PDP11/70

MEMORY TEST

MD-11-DEMJA-B

EP DEMJA B DL A

OCT 1976

FICHE 1 OF 1

digital

Made In U.S.A.

B01

```
$PS67890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012LPT
DBSTART*ROGer DAVIES,TOM [400,2284] SSSS DEMJAB SITITI2 D000 13Oct76 15:03:26 Monitor IPC-D S07B [1A3] *START*
D   D  A   A  V   V   I   E     S           T     O   O  MM MM
D   D  A   A  V   V   I   E     S           T     O   O  M M M
D   D  A   A  V   V   I   EEEE   SSS        T     O   O  M   M
D   D  AAAAA  V   V   I   E         S       T     O   O  M   M
D   D  A   A   V V    I   E         S  ..   T     O   O  M   M
DDDD   A   A    V    III  EEEEE SSSS    .   T      OOO   M   M
```

```
MTO:ML21-4:DAVIES2-- distribution to ML21-4, slot 134     1      333   4   4     >
MM MM  L       2   2    11          4   4         <     11     3   3  4   4      >
M M M  L           2     1          4   4        <       1         3  4   4       >
M   M  L          2      1   -----  44444       <        1        3   44444        >
M   M  L         2       1              4        <       1         3      4       >
M   M  L        2        1              4         <      1     3   3      4      >
M   M  LLLLL   22222    111             4          <    111     333       4     >
```

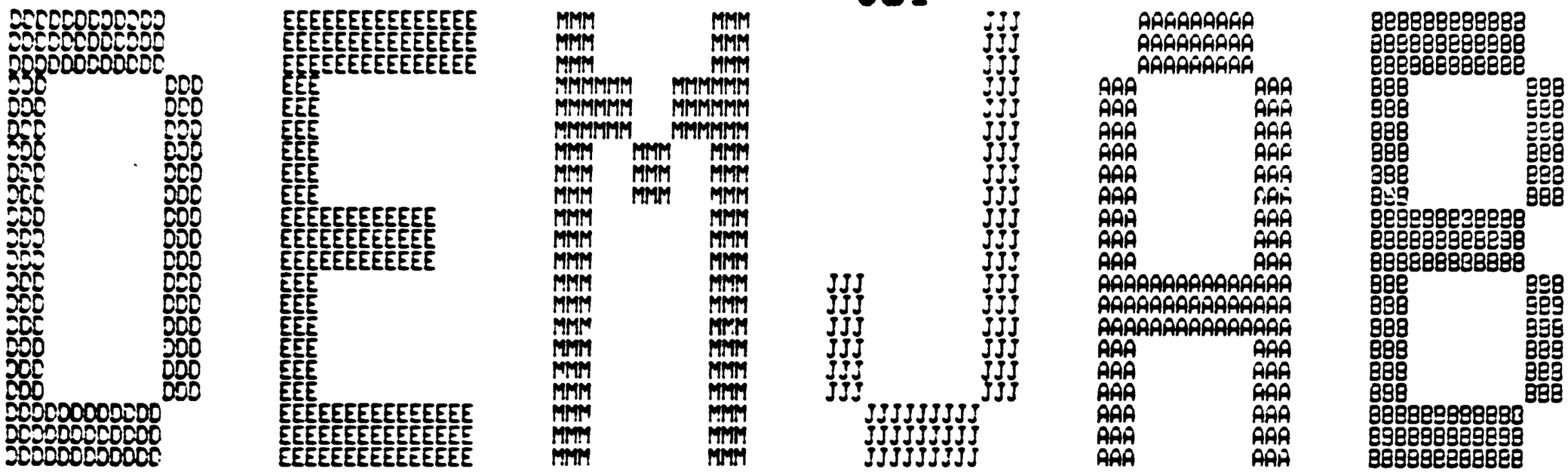
```
/DBSTART*ROGer DAVIES,TOM [400,2284] SSSS DEMJAB SITITI2 Date 13-Oct-76 15:03:26 Monitor IPC-D S07B [1A3] *START*
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000011111111111111111111111111111111110
000000000111111111122222222223333333333444444444455555555556666666666777777777788888888889999999999000000000011111111112222222222333312
123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012 *S
```

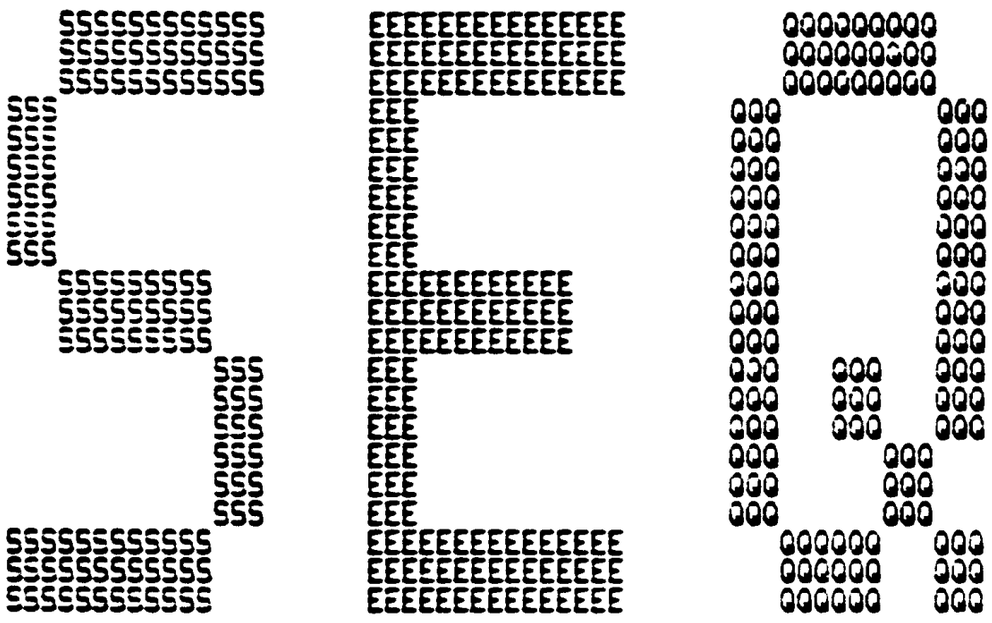

```
D   D  A   A  V   V   I   E     S           T     O   O  MM MM
D   D  A   A  V   V   I   E     S           T     O   O  M M M
D   D  A   A  V   V   I   EEEE   SSS        T     O   O  M   M
D   D  AAAAA  V   V   I   E         S       T     O   O  M   M
D   D  A   A   V V    I   E         S  ..   T     O   O  M   M
DDDD   A   A    V    III  EEEEE SSSS    .   T      OOO   M   M
```

```
MTO:ML21-4:DAVIES2-- distribution to ML21-4, slot 134     1      333   4   4     >
MM MM  L       2   2    11          4   4         <     11     3   3  4   4      >
M M M  L           2     1          4   4        <       1         3  4   4       >
M   M  L          2      1   -----  44444       <        1        3   44444        >
M   M  L         2       1              4        <       1         3      4       >
M   M  L        2        1              4         <      1     3   3      4      >
M   M  LLLLL   22222    111             4          <    111     333       4     >
```

```
/DBSTART*ROGer DAVIES,TOM [400,2284] SSSS DEMJAB SITITI2 Date 13-Oct-76 15:03:26 Monitor IPC-D S07B [1A3] *START*
```

C01

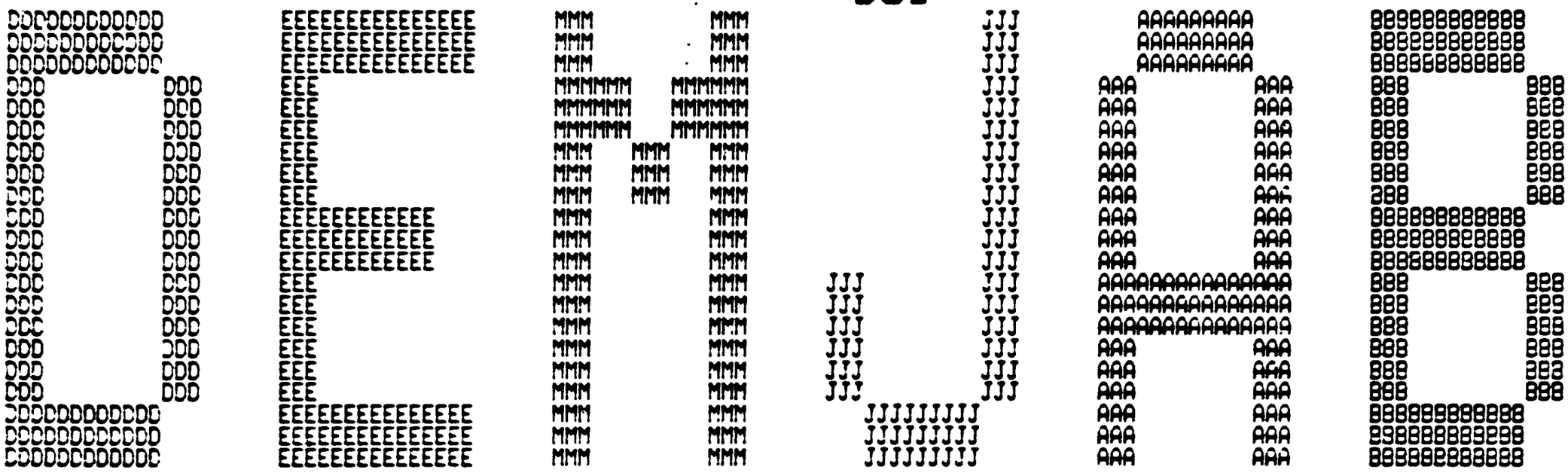


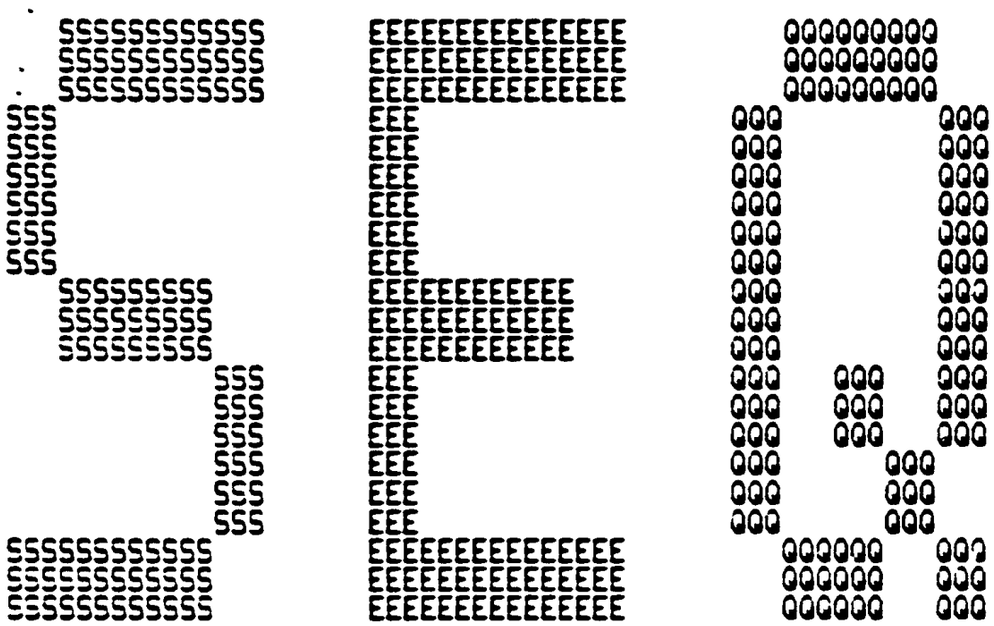


```
LPTSPL Version 6(100344)  Running on MTA1
 *START* User DAVIES,TOM [400,2704] Job  DEMJAB Seq. 12 Date 13-Oct-76 15:03:26 Monitor IPC-D 5078 [1A3] *START*
Request created: 13-Oct-76  14:48:15     /TO:ML21-4:DAVIES -- distribution to ML21-4, slot 134

[illegible]15:03:29
File will be RENAMEd to <057> protection
```

DO1





LPTSPL Version 6(100344) Running on MTA1
START User DAVIES,TOM [400,2704] Job DEMJAB Seq. 12 Date 13-Oct-76 15:03:26 Monitor IPC-D 5078 [1A3] *START*
Request created: 13-Oct-76 14:48:15 /TO:ML21-4:DAVIES -- distribution to ML21-4, slot 134

[illegible]
File will be RENAMEd to <057> protection

```
                                .REM    %
```

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DEMJA-B-D
PRODUCT NAME:	PDP-11/70 MEMORY TEST
DATE CREATED:	MARCH 1976
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	JIM LACEY

TABLE OF CONTENTS

1.0 ABSTRACT

PROGRAM DEMJA TESTS CONTIGUOUS MEMORY ADDRESS FROM 000000 TO 17757776. IT VERIFIES THAT EACH ADDRESS IS UNIQUE (AN ADDRESS TEST) AND THAT EACH MEMORY LOCATION CAN BE READ/WRITTEN RELIABLY (WORST CASE NOISE TESTS).
THIS PROGRAM MAY BE USED TO ADJUST/MARGIN MEMORY.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11/70 FAMILY PROCESSOR WITH 32K MEMORY

2.2 STORAGE

PROGRAM STORAGE - THE PROGRAM USES MEMORY 0-17777

2.3 PRELIMINARY PROGRAMS

DEKBA THROUGH DEKBF

3.0 LOADING AND STARTING PROCEDURE

LOAD PROGRAM INTO MEMORY USING ABS LOADER
LOAD ADDRESS 200
SET SW12 IN DESIRED POSITION (SEE SEC 4.0)
PRESS START.
ASTERISK "*" WILL BE PRINTED AFTER EACH PASS.
"DEMJA DONE!" WILL BE PRINTED AFTER 6 PASSES.
PASS COUNT MAY BE MONITORED IN THE DISPLAY REGISTER.
NOTE: THIS PROGRAM SAVES THE LOADERS (BOOT AND ABS), TO RESTORE THE LOADERS, RESTART AT 162.

3.1 ACT11 OPERATION

IF THE PROGRAM IS RUN IN QUICK VERIFY MODE UNDER ACT11 THE PROGRAM IS DONE AFTER THE FIRST PASS..PAGE

4.0 SWITCH SETTINGS

SW15 = 1 OR UP.... HALT ON ERROR
NOTE: IF SW15=1 WHEN AN ERROR OCCURS THE PROGRAM WILL HALT, AND THE CORRECT DATA WILL NOT BE LOADED INTO THE FAILING ADDRESS. IF SW15 IS RAISED AFTER THE ERROR TYPEOUT BEGINS THE PROGRAM WILL HALT WHEN THE TYPEOUT COMPLETES, AND THE CORRECT DATA WILL BE LOADED INTO THE FAILING ADDRESS.

SW14 = 1 OR UP.... LOOP SUBTEST

SW13 = 1 OR UP..... INHIBIT ERROR TYPEOUT

SW12 = 1 OR UP....INHIBIT USE OF MEMORY MANAGEMENT
NOTE: INHIBITING THE USE OF MEMORY MANAGEMENT CAN

```
   151                                                  BE DONE ONLY WHEN THE PROGRAM IS STARTED.
   152                                                  IF THE USE OF MEMORY MANAGEMENT IS INHIBITED THE  LAST
```

ADDRESS AS TYPED BY THE PROGRAM WILL ONLY REFLECT THE AMOUNT OF MEMORY UP TO 28K (LAST ADDRESS = 160000).

SW11 = 1 OR UP..... INHIBIT SUBTEST ITERATION

SW10 = 1 OR UP..... RING BELL ON ERROR

SW9 = 1 OR UP..... DISPLAY ERROR COUNT IN DISPLAY REGISTER

SW9 = 0 OR DOWN... DISPLAY PASS COUNT IN DISPLAY REGISTER

SW8 = 1 OR UP..... HALT PROGRAM UNRELOCATED & RESTORE LOADERS.

5.0 SUBROUTINE ABSTRACTS

5.1 SCOPE

THE PROGRAM STORES IN R1 THE PC OF THE LAST TEST SUCCESSFULLY EXECUTED AND MAY BE USED AS AN AID IN DEBUGGING IF THE PROGRAM 'BOMBS' BECAUSE OF A HARDWARE FAILURE.

6.0 ERRORS

THESE TESTS PRINT OUT THE PC WHERE THE ERROR WAS DETECTED, THE FAILING ADDRESS, THE GOOD DATA, AND THE BAD DATA I.E.

PC=XXXXXX ADDRESS AAAAAA GOOD DATA GGGGGG BAD DATA BBBBBB

THE ADDRESS OF THE FAILING LOCATION IS THE TRUE 22 BIT PHYSICAL ADDRESS.

NOTE: WHEN TESTING MEMORY LOCATIONS 0-77776 THE PC TYPED WILL BE A MULTIPLE OF 100000 GREATER THAN RELFECTED IN THE PROGRAM LISTING
THE ADDRESS OF THE BAD DATA IS IN (R2) -2
THE GOOD DATA IN R0
THE BAD DATA IN R3
THE ADDRESS OF GOOD DATA IS IN R4 (RANDOM DATA TEST ONLY)
WHEN AN ERROR IS DETECTED WHEN EXERCISING THE MEMORY USING THE WORST CASE NOISE PATTERNS, THE USER SHOULD RESTART THE PROGRAM SELECTING PROGRAM #2(SEE SEC 9.1 FOR DETAILS) SELECTING THE APPROPRIATE PARAMETERS. THE USER CAN USE THE PC AND ADDRESS OF THE FAILURE TO SELECT THE PROPER CORE BANK(S) AFFECTED AND ALSO THE SPECIFIC PATTERN. THIS ALLOWS MAXIMUM SCOPE CAPABILITIES.

6.1 PARITY ERROR

IF A PARITY ERROR IS DETECTED THE PROGRAM WILL TYPE:

PARITY ERROR

```
                                        PC=PPPPPP MEMORY ADDRESS IS AAAAAAAA
                                        PARITY ERROR REG=EEEEEE  ?????????? MARGIN
```

WHERE PPPPPP IS THE CONTENTS OF THE PC WHEN THE PARITY ERROR OCCURRED, AAAAAAAA IS THE ADDRESS OF THE WORD, EEEEEE IS THE CONTENTS OF THE MEMORY ERROR REGISTER, AND ????????? IS THE MARGIN SETTING AT THE TIME OF THE PARITY ERROR.

AFTER REPORTING THE PARITY ERROR THE PROGRAM WILL START OVER.

7.0 RESTRICTIONS

7.1 STARTING RESTRICTION

PROGRAM MUST NOT BE RELOCATED WHEN RESTARTING

7.2 OPERATIONAL RESTRICTION

PROGRAM CHECKS CONTIGUOUS MEMORY IF A PARITY ERROR TRAP OCCURS WHEN THE PROGRAM IS RELOCATED PROGRAM ACTION IS UNDEFINED. IF PARITY MEMORY IS AVAILABLE OR SELECTED THE 3XOR9 TEST PATTERN IS FOR PARITY MEMORY ONLY. DO NOT POWER FAIL THE PROGRAM WHEN THE PROGRAM IS RUNNING RELOCATED.

8.0 MISCELLANEOUS

IF THE PROGRAM HALTS IN THE TRAP/INTERRUPT VECTOR AREA (0-1000), EXAMINE REGISTER 6 (THE STACK PTR). R6 CONTAINS THE ADDRESS WHERE THE PC OF THE INSTRUCTION THAT CAUSED THE TRAP ABORT IS STORED. SEE ALSO R1 (R1 SPECIFIES THE LAST TEST COMPLETED).

NOTE: THE PDP-11/?0 WILL DISPLAY THE TRAP VECTOR ADDRESS+4 IN THE ADDRESS LIGHTS. THUS A TRAP TO 4 (BUS ERROR) WILL DISPLAY 10 IN THE ADDRESS LIGHTS.

8.1 STACK POINTER

THE STACK POINTER IS INITIALLY SET TO 520 AND IS RESET TO THIS VALUE AT THE START OF EACH SUBTEST.

8.2 PASS COUNT

SIX PASSES ARE REQUIRED FOR COMPLETION OF THIS PROGRAM; AT WHICH TIME AN "*" WILL BE PRINTED. THE PASS COUNT MAY BE OBSERVED BY TURNING THE SWITCH TO THE DISPLAY POSITION. (THE PASS COUNT IS ALSO STORED IN LOCATION 1000.) THE PASS COUNT SHOULD BE MONITORED IN THE EVENT THAT THE PROGRAM ENTERS AN UNDEFINED LOOP..BLANK 1

8.3 ERROR COUNT

EACH TIME AN ERROR OCCURS, THE ERROR COUNT IS INCREMENTED. THE ERROR COUNT CAN BE OBSERVED BY TURNING THE SWITCH TO THE DISPLAY POSITION AND SETTING SWITCH 9. (THE ERROR COUNT IS ALSO STORED IN LOCATION 1002.) THE PROGRAM WILL COUNT 17777(8) ERRORS; THE ERROR COUNT IS NOT INCREMENTED PAST THIS VALUE..BLANK 1

8.4 DISPLAY REGISTER

EITHER THE PASS COUNT OR THE ERROR COUNT IS DISPLAYED IN THE DISPLAY REGISTER. THE COUNT TO BE DISPLAYED IS CONTROLLED BY THE SETTTING OF SWITCH 9..BLANK 1

```
267
268                                     8.5     POWER FAIL
```

```
                                THE PROGRAM MAY BE POWER FAILED WHEN RUNNING.  WHEN THE  POWER
                                RETURNS  THE  PROGRAM WILL CONTINUE IN SEQUENCE.**CAUTION** DO
                                NOT TURN POWER OFF/ON UNTIL THE  MESSAGE  'POWER  FAILED'  HAS
                                BEEN TYPED.  THIS IS BECAUSE THE STACK MAY OVERFLOW.

                        8.6     EXECUTION TIME
                                EXECUTION TIME IS DEPENDENT ON THE AMOUNT OF MEMORY.

                        9.0     PROGRAM DESCRIPTION
                                THE PROGRAM VERIFIES EACH ADDRESS BY WRITING THE VALUE OF EACH
                                ADDRESS  INTO  ITSELF STARTING AT LOCATION 20000 AND ENDING AT
                                THE LAST LOCATION IN MEMORY.  THE VALUE OF THE  LAST  LOCATION
                                +2 IS TYPED ON THE TTY.  NEXT THE VALUES WRITTEN ARE VERIFIED.
                                TO COMPLETE THE ADDRESS TEST  THE  COMPLEMENT  VALUE  OF  EACH
                                MEMORY  ADDRESS IS WRITTEN STARTING AT THE LAST MEMORY ADDRESS
                                AND ENDING AT ADDRESS 20000.  THE  WRITTEN  COMPLEMENT  VALUES
                                ARE  THEN  VERIFIED.   THE  NEXT  PHASE  OF  TESTING  INCLUDES
                                READING, WRITING AND CHECKING MEMORY USING  WORST  CASE  NOISE
                                TEST PATTERN.  A SUBTEST IS DEDICATED TO CHECKING THE PATTERN.
                                THE TEST PROCEEDS BY EXERCISING EACH BANK OF MEMORY USING  THE
                                WORST  CASE  PATTERN.   THE  PROGRAM  THEN CHECKS MEMORY USING
                                RANDOM DATA (RANTST).  THIS ROUTINE  MOVES  THE  PROGRAM  CODE
                                THROUGHOUT  MEMORY  STARTING  AT LOCATION 20000, AND RELOCATES
                                THE  DATA  BY  A  32(10)  WORD  OFFSET  ON   EACH   SUBSEQUENT
                                RELOCATION.   I.E.,  FIRST  RELOCATION IS TO 20000, NEXT IS TO
                                20100, THEN 20200, ETC.  AFTER RELOCATION THE  CODE  MOVED  IS
                                CHECKED  AGAINST THE ORIGINAL CODE (0-17776).  WHEN THE RANDOM
                                DATA  ST IS COMPLETE THE PROGRAM THEN SUCCESSIVELY ROTATES  A
                                0 BI  (ROT0) AND A '1' BIT (ROT1) THROUGH ALL OF MEMORY.  WHEN
                                ALL TESTING IS COMPLETE THE PROGRAM INCREMENTS THE PASS  COUNT
                                (LOCATION  1000)  AND  RESTARTS  BEGINNING WITH THE WORST CASE
                                NOISE TESTS.  AN ASTERISK (*) WILL BE TYPED ON  COMPLETION  OF
                                EACH  PASS,  AND WHEN 6 PASSES HAVE BEEN COMPLETED THE PROGRAM
                                WILL TYPE 'DEMJA DONE' AND RESTART THE PROGRAM BEGINNING  WITH
                                THE MEMORY ADDRESS TESTS.
                                %
                                .NLIST  MD,MC
                                .LIST   ME
                                .ABS
                                .MCALL  $TYPE
                                .TITLE  MAINDEC-11-DEMJA-B PDP11/70 MEMORY TEST
                                .SBTTL  STARTING INST & DEFINITIONS
                        ;COPYRIGHT 1973 1976 DIGITAL EQUIPMENT CORP., MAYNARD,MASS.

                        ;THIS TEST CHECKS THAT ALL MEMORY ADDRESSES ARE UNIQUE USING ADDRESS TESTS
                        ;AND CHECKS DATA RELIABILITY OF MEMORY USING WORST CASE NOISE TEST PATTERNS
                        ;A RANDOM # PATTERN (PROGRAM CODE RELOCATED), A ROTATING 0 AND ROTATING
                        ;1 PATTERN.

                        ;LOADING AND STARING INSTRUCTIONS
                        ;LOAD ADDRESS 200 AND START
                        ;NOTE:  PROGRAM WILL RUN WORST CASE TEST PATTERNS IN LOWEST 4K
                        ;THUS THE PROGRAM CANNOT BE RESTARTED AT 200 IF  _OCATED. TO PREVENT
```

```
                                ;RELOCATION FROM OCCURING DEPOSIT 200 INTO LOCATION 42 (NOT NECESSARY
                                ;IF LOADED VIA ACT11). THIS ACTION WILL PREVENT RELOCATION AND ALSO
                                ;INHIBIT TESTING MEMORY IN LOWEST 4K.
                                ;THIS PROGRAM ALSO RELOCATES THE ABS AND BOOT LOADERS TO ALLOW TESTING
                                ;OF MEMORY, TO RESTORE THE LOADERS RESTART AT 162.
                                ;       STACK POINTER IS SET AT 500
                                ;       AN ASTERISK '*' WILL BE PRINTED ON COMPLETION OF EACH PASS, AND
                                ;       THE PROGRAM NAME WILL BE PRINTED WHEN TEST IS COMPLETE.

                                ;GENERAL REGISTER ASSIGNMENTS
000000                                  R0=%0
000001                                  R1=%1
000002                                  R2=%2
000003                                  R3=%3
000004                                  R4=%4
000005                                  R5=%5
000006                                  SP=%6
000007                                  PC=%7
000000                                  R10=%0
000001                                  R11=%1
000002                                  R12=%2
000003                                  R13=%3
000004                                  R14=%4
000005                                  R15=%5

                                ;STATUS REGISTER (PSW) BIT ASSIGNMENTS
000001                                  C=1                     ;C BIT
000002                                  V=2                     ;V BIT
000004                                  Z=4                     ;Z BIT
000010                                  N=10                    ;N BIT
000020                                  T=20                    ;'T' BIT
000340                                  PRTY7=340               ;PRIORITY LEVEL 7
000200                                  PRTY4=200               ;PRIORITY LEVEL 4
000000                                  KM=000000               ;KERNEL MODE
040000                                  SM=040000               ;SUPERVISORY MODE
140000                                  UM=140000               ;USER MODE
000000                                  PKM=000000              ;PREVIOUS KERNEL MODE
010000                                  PSM=010000              ;PREVIOUS SUPERVISORY MODE
030000                                  PUM=030000              ;PREVIOUS USER MODE
004000                                  REG=004000              ;SELECT R10-R15

                                ;VECTOR ADDRESSES
000004                                  ERRVEC=4                ;ADDRESS OF ERROR VECTOR
000010                                  RESVEC=10               ;ADDRESS OF RESERVED INST. TRAP VECTOR
000014                                  TBITVEC=14              ;ADDRESS OF 'T' BIT TRAP VECTOR
000014                                  TRTVEC=14               ;ADDRESS OF 'TRACE' TRAP VECTOR
000014                          BPTVEC=14               ;ADDRESS OF 'BREAKPOINT' TRAP VECTOR
000020                                  IOTVEC=20               ;ADDRESS OF IOT TRAP VECTOR
000024                                  PFVEC=24                ;ADDRESS OF POWER FAIL TRAP VECTOR
000030                                  EMTVEC=30               ;ADDRESS OF EMT VECTOR
000034                                  TRAPVEC=34              ;ADDRESS OF TRAP VECTOR
000060                                  TKVEC=60                ;ADDRESS OF TTY KEYBOARD INTERRUPT VECTOR
000064                                  TPVEC=64                ;ADDRESS OF TTY PRINTER INTERRUPT VECTOR
000240                                  PIRVEC=240              ;ADDRESS OF PIRQ VECTOR
000244                                  FPEVEC=244              ;ADDRESS OF FLOATING POINT INT. VECTOR
000250                                  MMVEC=250               ;ADDRESS OF MEM MGMT ERROR TRAP VECTOR
```

```
                                ;REGISTER ADDRESSES
            177776                      PSW=177776          ;ADDRESS OF STATUS REGISTER
            177774                      SLR=177774          ;ADDRESS OF STACK LIMIT REGISTER
            177772                      PIRQ=177772         ;ADDRESS OF PROGRAM INTERRUPT REQUEST
            177770                      UBREAK=177770       ;ADDRESS OF MICRO BREAK REGISTER
            177560                      TKS=177560          ;ADDRESS OF KEYBOARD CSR
            177562                      TKB=177562          ;ADDRESS OF KEYBOARD BUFFER
            177564                      TPS=177564          ;ADDRESS OF TELEPRINTER CSR
            177566                      TPB=177566          ;ADDRESS OF TELEPRINTER BUFFER
            177570                      SWR=177570          ;ADDRESS OF CONSOL SWITCH REGISTER
            177570                      DISPLAY=177570      ;ADDRESS OF CONSOL DISPLAY REGISTER

                                ;INITIAL STACK POINTER SETTING
            000500                      STKPTR=500

                                ;MISCELLANEOUS BIT ASSIGNMENTS
            000100                      BIT15=  100
            040000                      BIT14=  040000
            020000                      BIT13=  020000
            010000                      BIT12=  010000
            001000                      BIT9=   001000
            000400                      BIT8=   000400
            000100                      BIT6=   000100

                                ;MEMORY MANAGEMENT REGISTER ADDRESS ASSIGNMENTS
            177572                      SR0=177572          ;ADDRESS OF MEM MGMT REGISTER SR0
            177574                      SR1=177574          ;    "    "    "    "    "    SR1
            177576                      SR2=177576          ;    "    "    "    "    "    SR2
            172516                      SR3=172516          ;ADDRESS OF MEM MGMT REGISTER SR3

            172300                      KIPDR0=172300       ;ADDRESS OF KERNEL 'I' PAGE
            172302                      KIPDR1=172302       ;DESCRIPTOR REGISTERS
            172304                      KIPDR2=172304
            172306                      KIPDR3=172306
            172310                      KIPDR4=172310
            172312                      KIPDR5=172312
            172314                      KIPDR6=172314
            172316                      KIPDR7=172316

            172340                      KIPAR0=172340       ;ADDRESSES OD KERNEL 'I' SPACE
            172342                      KIPAR1=172342       ;PAGE ADRESS REGISTERS
            172344                      KIPAR2=172344
            172346                      KIPAR3=172346
            172350                      KIPAR4=172350
            172352                      KIPAR5=172352
            172354                      KIPAR6=172354
            172356                      KIPAR7=172356


                                ;INSTRUCTION EQUATES
            104400                      HLT=TRAP
            104000                      SCOPE=EMT           ;SCOPE IS AN EMT TRAP

                                ;MISC. EQUATES
            000006                      RW=6                ;R/W BIT IN PDR REGISTERS
```

```
                    000000                          UP=0                    ;UP BIT IN PDR REGISTERS
                    000000                          .=0
000000  000000                                     .WORD   0               ;SPECIAL TRAP/INTERRUPT CATCHER IF PRO-
000002  000000                                     .WORD   0               ;GRAM HALTS AT 0 THEN ADDRESS WAS NOT
                                                                            ;LOADED PROPERLY FROM VECTOR.
000004  001126                                     .WORD   ERRTRP
000006  000002                                     .WORD   RTI
                    000034                          .=TRAPVEC
000034  001204                                     .WORD   ERROR
000036  000340                                     .WORD   PRTY7
                    000046                          .=46
000046  004236                                    LOGICAL

                    000052                          .=52
000052  040000                                    40000
                    000100                          .=100
000100  004567  000664                 CRLF:      JSR     R5,$PRINT
000104  000746                                    $CRLF
000106  000207                                    RTS     PC
000110  000000                         RELFL:     .WORD   0
000112  000000                         SAVPC2:    .WORD   0
                    000162                          .=162
000162  012706  000500                 PONE:      MOV     #500,SP ;STARTING ADDRESS TO RELOCATE LOADERS.
000166  004767  002016                             JSR     PC,$RLDR
000172  000000                                    HALT
000174  000401                                    BR      PTWO
                    000200                          .=200
000200  012706  000500                 PTWO:      MOV     #500,SP ;STARTING ADDRESS OF MEMORY TEST.
000204  000137  002376                             JMP     @#START         ;GO TO START OF TEST
                    000250                          .=250
000250  000000                                     .WORD   0       ;MEMORY MANAGEMENT TRAP VECTOR.
000252  000000                                     .WORD   0
                                                ;
                                                ;ROUTINE TO SAVE REGISTERS ON THE STACK
                                                ;CALLED BY SAVE MACRO OR JSR    PC,$SAVR
000254  012667  000016                 $SAVR:     MOV     (SP)+,1$        ;SAVE RETURN PC
000260  010546                                    MOV     R5,-(SP)
000262  010446                                    MOV     R4,-(SP)
000264  010346                                    MOV     R3,-(SP)
```

```
                                                MOV     R2,-(SP)
                                                MOV     R1,-(SP)
                                                MOV     R0,-(SP)
                                                MOV     (PC)+,PC        ;RETURN
                                        1$:     0                       ;CONTAINS RETURN ADDRESS

                                        ;ROUTINE TO RESTORE REGISTERS SAVED ON THE STACK
                                        ;CALLED BY RESTORE MACRO OR JSR  PC,$RESTR
                                        $RESTR: MOV     (SP)+,1$        ;SAVE RETURN PC
                                                MOV     (SP)+,R0
                                                MOV     (SP)+,R1
                                                MOV     (SP)+,R2
                                                MOV     (SP)+,R3
                                                MOV     (SP)+,R4
                                                MOV     (SP)+,R5
                                                MOV     (PC)+,PC        ;RETURN
                                        1$:     0                       ;CONTAINS RETURN ADDRESS

                                                .SBTTL  POWER FAIL ROUTINE
                                                .=502
                                        ;POWER FAIL ROUTINE
                                        ;THE POWER DOWN ROUTINE SAVES THE KEYBOARD STATUS,THE GENERAL REGISTERS
                                        ;(R0-R5) AND MEM MGMT REGISTERS (KIPDR0-KIPDR7,KIPAR0-KIPAR7,SR3,SR2,SR0)
                                        ;ON THE STACK AND SAVES THE STACK POINTER IN PFSTK BELOW.
                                        PDWN:   MOV     @#TKS,-(SP)     ;SAVE KEYBOARD STATUS
                                                JSR     PC,$SAVR        ;GO SAVE REGISTERS ON THE STACK
                                                TST     @#MMAVA         ;CHECK IF MEM MGMT IS AVAILABLE
                                                BEQ     3$              ;BRANCH IF NOT AVAILABLE
                                                MOV     @#SR0,-(SP)     ;SAVE SR0
                                                MOV     @#SR2,-(SP)     ;SAVE SR2
                                                MOV     @#SR3,-(SP)     ;SAVE SR3
                                                MOV     #KIPDR0,R0      ;GET ADDRESS OF KIPDR0
                                                MOV     #8.,R2
                                                MOV     R2,R3
                                        1$:     MOV     (R0)+,-(SP)     ;SAVE KIPDR0-KIPDR7
                                                SOB     R2,1$
                                                MOV     #KIPAR0,R0      ;GET ADDRESS OF KIPAR0
                                        2$:     MOV     (R0)+,-(SP)     ;SAVE KIPAR0-KIPAR7
                                                SOB     R3,2$
                                        3$:     MOV     SP,(PC)+        ;SAVE STACK PTR IN FOLLOWING LOCATION
                                        PFSTK:  .WORD   0               ;CONTAINS STACK PTR AFTER POWER FAIL
                                                MOV     #PUP,@#PFVEC    ;SET POWER FAIL VECTOR TO PUP ROUTINE
                                                HALT

                                        ;POWER UP ROUTINE.
                                        PUP:    NOP
                                                MOV     @#PFSTK,SP      ;SET STACK PTR
                                                TST     MMAVA           ;CHECK IF MEM MGMT IS AVAILABLE
                                                BEQ     4$
                                                MOV     #KIPAR7+2,R0    ;GET ADDRESS OF KIPAR7+2
                                                MOV     #8.,R2
                                                MOV     R2,R3
                                        1$:     MOV     (SP)+,-(R0)     ;RESTORE KIPAR7-KIPAR0
                                                SOB     R3,1$
                                                MOV     #KIPDR7+2,R0    ;GET ADDRESS OF KIPDR7+2
                                        2$:     MOV     (SP)+,-(R0)     ;RESTORE KIPDR7-KIPDR0
```

```
                                                SUB     R2,2$
                                                MOV     (SP)+,@#SR3     ;RESTORE SR3
                                                MOV     (SP)+,@#SR2     ;RESTORE SR2
                                                MOV     (SP)+,@#SR0     ;RESTORE SR0
                                        4$:     TST     PARAVA          ;CHECK IF PARITY REGISTERS ARE ENABLED
                                                BEQ     5$              ;BRANCH IF NOT
                                                JSR     PC,..MAMF       ;GO ENABLE PARITY REGISTERS
                                        5$:
                                                JSR     PC,$RESTR       ;RESTORE REGISTERS FROM STACK
                                                MOV     (SP)+,@#TKS
                                                MOV     #PDWN,@#PFVEC   ;SET POWER FAIL TRAP TO PDWN ROUTINE
                                                CLR     (PC)+
                                        10$:    .WORD   0
                                        11$:    INC     10$             ;DELAY WAITING FOR TTY MOTOR
                                                BPL     11$
                                                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
                                                PWRFAIL
                                        6$:     NOP
                                                RTI                     ;RETURN

                                        PWRFAIL:.ASCII  <15><12>'POWER FAILED'
                                        $CRLF:  .ASCIZ  <15><12>


                                                .SBTTL  TAGS & PRINT ROUTINE
                                                .EVEN
                                        ICNT:   .WORD   0                       ;CONTAINS PASS COUNT
                                        ICOUNT: .WORD   0               ;CONTAINS ITERATION PATTERN
                                        ERCNT:  0                       ;CONTAINS ERROR COUNT
                                        LDDISP: 0                       ;CONTAINS DISPLAY REGISTER IMAGE
                                        MMAVA:  0                       ;MEM MGMT AVAILABLE INDICATOR
                                                                        ;0=NOT AVAIL,-1=AVAIL(18 BIT MODE)
                                                                        ;-2=AVAIL(22 BIT MODE)

                                        RELOCF: .WORD   0               ;CONTAINS RELOCATION FACTOR
                                        COUNT:  .WORD   0               ;TEMPORARY WORKING LOACATION

                                        ;ROUTINE TO PASS MESSAGE ADDRESS TO TYPE ROUTINE BELOW
                                        ;CALL:  JSR     R5,$PRINT
                                        ;       MESSAGE ADDRESS
                                        $PRINT: NOP
                                                MOV     (R5)+,1$        ;GET MESSAGE ADDRESS
                                                ADD     RELOCF,1$       ;ADD RELOCATION FACTOR
                                                MOV     @#PSW,-(SP)     ;PUSH PSW ON THE STACK
                                                JSR     PC,..TYPE       ;CALL TYPE ROUTINE
                                        1$:     .WORD   0               ;CONTAINS MESSAGE ADDRESS
                                                RTS     R5              ;RETURN

                                        ;ROUTINE TO TYPE ASCII MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                        ;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                        ;CALL:  TYPE
                                        ;               MESADR          ;MESADR IS FIRST ADDRESS OF ASCIZ STRING

                                        ;TAGS USED BY THE TYPE ROUTINE BELOW
```

```
                                        $NULL:  .BYTE   0               ;CONTAINS NULL CHARACTER
                                        $FILL:  .BYTE   2               ;CONTAINS # OF FILLER CHARACTERS
                                        $TPFLG: .BYTE   0               ;CONTAINS TELEPRINTER AVAILABLE FLAG
                                                                        ;0/377 = AVAIL/NOT AVAIL
                                        $TKFLG: .BYTE   0               ;CONTAINS KEYBOARD AVAILABLE FLAG
                                        $TPS:   .WORD   177564          ;ADDRESS OF TLEPRINTER STATUS REGISTER
                                        $TPB:   .WORD   177566          ;ADDRESS OF TELEPRINTER DATA BUFFER
                                        .TYPE:  MOV     R0,-(SP)        ;SAVE R0
                                                MOV     @2(SP),R0       ;GET MESSAGE ADDRESS
                                                ADD     #2,2(SP)        ;ADJUST RETURN PC

                                        1$:     MOVB    (R0)+,-(SP)     ;PUSH CHARACTER TO BE TYPED ONTO STACK
                                                BNE     2$              ;BRANCH IF NOT THE TERMINATOR
                                                TST     (SP)+           ;POP TERMINATOR CHAR OFF THE STACK
                                                MOV     (SP)+,R0        ;RESTORE R0
                                                RTI                     ;RETURN TO CALLER

                                        2$:     JSR     PC,TYPIT        ;TYPE CHARACTER
                                        3$:     CMPB    #12,(SP)+       ;CHECK IF CHARACTER WAS A LINE FEED
                                                BNE     1$              ;BRANCH IF NOT LINE FEED
                                                MOV     $NULL,-(SP)     ;GET # OF FILLERS REQUIRED AND FILLER
                                                                        ;CHARACTER.

                                        4$:     DECB    1(SP)           ;DECREMENT FILLERS REQ. COUNT
                                                BLT     3$              ;BRANCH IF NO MORE FILLERS ARE REQUIRED
                                                JSR     PC,TYPIT        ;TYPE FILLER CHARACTER
                                                BR      4$

                                        TYPIT:  TSTB    @$TPS           ;WAIT FOR OUTPUT DEVICE
                                                BPL     .-4
                                                MOVB    2(SP),@$TPB     ;OUTPUT CHARACTER
                                                RTS     PC

                                        ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
                                        ;ERROR TRAP SERVICE ROUTINE
                                        ERRTRP: TST     @#SWR           ;CHECK IF HALT ON ERROR
                                                BPL     .+4             ;BRANCH IF NO HALT ON ERROR
                                                HALT                    ;HALT
                                                TST     (PC)+           ;CHECK IF PREV TRAP TO 4 REPORTED
                                        1$:     .WORD   0               ;CONTAINS ERROR REPORTED FLAG
                                                BNE     2$              ;BRANCH IF NOT REPORTED
                                                MOV     SP,1$           ;SET 'NOT REPORTED'
                                                MOV     (SP),R2         ;GET PC OFF STACK
                                                JSR     PC,$FORMO       ;GO TO FORMAT ROUTINE
                                                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
                                                TRAP4
                                                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
                                                DIGITS
                                        2$:     HALT                    ;ERROR! SECOND TRAP TO 4 OCCURRED
                                                                        ;BEFORE FIRST WAS PRINTED
                                                CLR     1$
                                                JMP     @#200           ;RESTART AT 200

                                                .SBTTL  ERROR SERVICE ROUTINE
                                        ;ERROR SERVICE CALLED BY JSR  PC,ERROR INSTRUCTION
                                        ;OR HLT (A TRAP INST)
```

```
                                                ERROR:  NOP
                                                        CMP     #17777,ERCNT    ;CHECK FOR MAX ERROR CNT
                                                        BEQ     4$
                                                        ADD     #1,ERCNT        ;INCREMENT ERROR COUNT
                                                4$:     BIT     #BIT9,@#SWR     ;SWITCH 9 UP?
                                                        BEQ     5$
                                                        BIC     #17777,LDDISP   ;SAVE RELOCATION BITS
                                                        BIS     ERCNT,LDDISP    ;LOAD ERROR COUNT
                                                        MOV     LDDISP,@#DISPLAY        ;LOAD DISPLAY REGISTER
                                                5$:     TST     @#SWR           ;HALT ON ERROR
                                                        BPL     .+6
                                                        HALT
                                                        BR      3$
                                                        BIT     #20000,@#SWR    ;PRINT OUT DESIRED?
                                                        BNE     1$              ;BRANCH IF NO PRINTOUT
                                                        JSR     PC,$SAVR        ;GO SAVE REGISTERS ON THE STACK
                                                        MOV     14(SP),R2       ;GET PC OF ERROR CALL
                                                        JSR     PC,$FORMO       ;GO TO FORMAT ROUTINE
                                                        JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
                                                        ERRPC
                                                        JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
                                                        DIGITS
                                                        MOV     4(SP),R2        ;GET FAILING ADDRESS (IN R2)
                                                        JSR     PC,$FORMO       ;GO TO FORMAT ROUTINE
                                                        JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
                                                        ADRESS
                                                        TSTB    PENFLG          ;BRANCH IF PARITY ERROR DETECTED
                                                        BNE     11$             ;BUT NOT FOUND
                                                        TSTB    PEFLG           ;BRANCH IF PARITY ERROR DETECTED
                                                        BNE     10$             ;BUT FOUND
                                                        JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
                                                        XMTDAT
                                                        MOV     R0,-(SP)        ;PUSH VALUE TO TYPED ONTO STACK
                                                        JSR     PC,02A          ;GO PRINT VALUE
                                                10$:
                                                        JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
                                                        RECDAT
                                                        MOV     R3,-(SP)        ;PUSH VALUE TO BE TYPED ONTO STACK
                                                        JSR     PC,02A
                                                11$:    JSR     PC,CRLF
                                                        JSR     PC,$RESTR       ;RESTORE REGISTERS FROM STACK
                                                1$:     BIT     #2000,@#SWR     ;RING BELL ON ERROR
                                                        BEQ     2$
                                                        JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
                                                        BELL
                                                2$:     TST     @#SWR           ;HALT AFTER PRINT OUT
                                                        BPL     .+4
                                                        HALT
                                                3$:     MOV     R0,-(R2)        ;RESTORE CORRECT DATA TO ADDRESS
                                                        ADD     #2,R2
                                                        RTI

                                                TRAP4:  .ASCII  'TRAPPED TO 4 '
                                                ERRPC:  .ASCIZ  'PC='
```

```
717  001501  000107  047517  020104  XMTDAT: .ASCIZ  'GOOD DATA='
718  001506  040504  040524  000075
719  001514  041040  042101  042040  RECDAT: .ASCIZ  ' BAD DATA='
720  001522  052101  036501  000
721  001527  000007  000000          BELL:   .ASCIZ  <7>
722  001531  000120  051101  052111  PARREG: .ASCIZ  /PARITY ERROR REG=/
723  001536  020131  051105  047522
724  001544  020122  042522  036507
725  001552  000000
726          001554                  .EVEN
727                                  ;ROUTINE TO PLACE ASCII VALUE OF AN ADDRESS IN TO ADDRESS MESSAGE
728  001554  066767  177204  000014  $FORM0: ADD     RELOCF,11$+2
729  001562  066767  177176  000152          ADD     RELOCF,41$+2
730  001570  004767  176460                  JSR     PC,$SAVR        ;GO SAVE REGISTERS ON THE STACK
731  001574  012704  002351          11$:    MOV     #DIGITS,R4      ;ADDRESS WHERE ASCII VALUES ARE STORED
732  001600  005003                  CLR     R3              ;WORKING & INDEX REGISTER
733  001602  162702  000002          SUB     #2,R2           ;ADJUST ADDRESS
734  001606  010205                  MOV     R2,R5           ;SAVE
735  001610  010501                  MOV     R5,R1
736  001612  005767  177144          TST     MMAVA           ;CHECK IF MEM MGMT IS AVAILABLE
737  001616  001426                  BEQ     1$              ;BRANCH IF NOT AVAILABLE
738  001620  032737  000001  177572  BIT     #1,@#SR0        ;IS MEM MGMT ENABLED
739  001626  001422                  BEQ     1$
740  001630  042701  017777          BIC     #17777,R1       ;SAVE PAR SELECTOR BITS
741  001634  000301                  SWAB    R1              ;SWAP BYTES
742  001636  006001                  ROR     R1
743  001640  006001                  ROR     R1              ;FORM INDEX VALUE
744  001642  006001                  ROR     R1
745  001644  006001                  ROR     R1
746  001646  017102  001774          MOV     @PARTAB(1),R2   ;GET CONTENTS OF PAR
747  001652  012700  000006          MOV     #6,R0           ;SHIFT COUNT
748  001656  006302                  ASL     R2              ;SHIFT KIPAR1 6 PLACES LEFT
749  001660  006103                  ROL     R3              ;MSB'S GO INTO R3
750  001662  077003                  SOB     R0,.-4
751  001664  042705  160000          BIC     #160000,R5      ;CLEAR PAR SELECTOR BITS
752  001670  060502                  ADD     R5,R2           ;FORM 22 BIT ADDRESS
753  001672  005503                  ADC     R3              ;IN R2 & R3
754  001674  005001          1$:     CLR     R1
755  001676  012700  000005          MOV     #5,R0
756  001702  006003          12$:    ROR     R3
757  001704  006002                  ROR     R2
758  001706  006001                  ROR     R1
759  001710  005300                  DEC     R0
760  001712  001373                  BNE     12$
761  001714  012700  000010          MOV     #8.,R0          ;DIGIT COUNT
762  001720  000405                  BR      3$              ;PRINT FIRST DIGIT
763  001722  006301          2$:     ASL     R1
764  001724  006102                  ROL     R2
765  001726  006103                  ROL     R3
766  001730  005305                  DEC     R5
767  001732  001373                  BNE     2$
768  001734  012705  000003          3$:     MOV     #3,R5           ;DIGIT SHIFT COUNT
769  001740  116324  002312          41$:    MOVB    DIGTAB(3),(4)+  ;LOAD DIGIT INTO MESSAGE
770  001744  005003                  CLR     R3              ;CLEAR INDEX
771  001746  005300                  DEC     R0              ;DEC DIGIT COUNT
772  001750  001364                  BNE     2$
```

```
                001752  004767  176322                  JSR     PC,$RESTR       ;RESTORE REGISTERS FROM STACK
                001756  046767  177002  177612          BIC     RELOCF,11$+2
                001764  046767  176774  177750          BIC     RELOCF,41$+2
                001772  000207                          RTS     PC              ;RETURN

                001774  172340          PARTAB: KIPAR0
                001776  172342                  KIPAR1
                002000  172344                  KIPAR2
                002002  172346                  KIPAR3
                002004  172350                  KIPAR4
                002006  172352                  KIPAR5
                002010  172354                  KIPAR6
                002012  172356                  KIPAR7

                                ;ROUTINE TO TYPE OCTAL VALUE PUSHED ONTO STACK
                                ;CALL:  MOV     VALUE,-(SP)     ;PUSH VALUE ONTO STACK
                                ;       JSR     PC,02A          ;CALL ROUTINE
                002014          02A:
                002014  004767  176234                  JSR     PC,$SAVR        ;GO SAVE REGISTERS ON THE STACK
                002020  016600  000016                  MOV     16(SP),R0       ;GET VALUE
                002024  012703  000006                  MOV     #6,R3           ;COUNTER
                002030  005002                          CLR     R2              ;WORKING REGISTER
                002032  006100                          ROL     R0
                002034  006102                          ROL     R2
                002036  062702  000260          1$:     ADD     #260,R2         ;FORM ASCII VALUE
                002042  010267  000040                  MOV     R2,2$           ;MOVE CHAR TO TYPE LOCATION
                002046  004567  176716                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
                002052  002106                          2$
                002054  005002                          CLR     R2
                002056  006100                          ROL     R0
                002060  006102                          ROL     R2
                002062  006100                          ROL     R0
                002064  006102                          ROL     R2
                002066  006100                          ROL     R0
                002070  006102                          ROL     R2
                002072  005303                          DEC     R3
                002074  001360                          BNE     1$
                002076  004767  176176                  JSR     PC,$RESTR       ;RESTORE REGISTERS FROM STACK
                002102  012616                          MOV     (SP)+,(SP)
                002104  000207                          RTS     PC
                002106  000000          2$:     .WORD   0               ;CONTAINS CHARACTER TO BE TYPED

                002110  000000          LODFLG: .WORD   0
                                ;ROUTINE TO SAVE ABS LOADER
                002112  005767  177772  $LDR:   TST     LODFLG
                002116  001401                  BEQ     3$
                002120  000207                  RTS     PC
                002122  012700  017776  3$:     MOV     #17776,R0
                002126  012737  002140  000004        MOV     #2$,@#ERRVEC    ;SET TIME OUT TRAP VECTOR
                002134  005720                  TST     (R0)+
                002136  000776                  BR      .-2
                002140  022626          2$:     CMP     (SP)+,(SP)+
                002142  022700  020000                  CMP     #20000,R0       ;4K MACHINE?
                002146  001417                  BEQ     4$              ;YES--GET OUT
                002150  162700  005672                  SUB     #1500.+1*2,R0   ;POINT R0 BACK TO LOADER
                002154  010067  000102                  MOV     R0,$LDR1        ;SAVE FOR RESTORE ROUTINE
```

```
                                        MOV     #1500.,R2       ;WORD COUNT
                                        MOV     #LODAR,R3       ;WHERE LOADER IS TO BE STORED
                                1$:     MOV     (R0)+,(R3)+     ;STORE LOADER
                                        DEC     R2
                                        BNE     1$
                                        MOV     -(R3),LSTLOC    ;SAVE LAST WORD OF LOADERS
                                        DEC     LODFLG
                                4$:     RTS     PC              ;RETURN

                                ;ROUTINE TO RESTORE LOADER
                                $RLDR:  TST     LODFLG
                                        BNE     2$
                                        RTS     PC
                                2$:     MOV     $LDR1,R5        ;GET FIRST ADDRESS OF WHERE LOADER IS
                                                                ;TO BE RESTORED
                                        MOV     #LODAR,R4       ;ADDRESS WHERE LOADER IS STORED
                                        MOV     #1500.,R2       ;WORD COUNT
                                1$:     MOV     (R4)+,(R5)+     ;RESTORE
                                        DEC     R2
                                        BNE     1$
                                        MOV     (PC)+,-(R5)     ;RESTORE LAST LOCATION (SAVED BY SAVE
                                LSTLOC: .WORD   0               ;LOADERS ROUTINE ABOVE)
                                        JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
                                        $LDRM
                                        CLR     LODFLG
                                        RTS     PC              ;RETURN TO CALLER

                                $LDR1:  .WORD   0               ;FIRST ADDRESS WHERE LOADERS ARETO BE
                                                                ;RESTORED TO
                                $LDRM:  .ASCIZ  'LOADER IS RESTORED'<15><12>

                                        .EVEN
                                ;DIGIT TABLE
                                DIGTAB: "01
                                        "23
                                        "45
                                        "67

                                ;MESSAGES
                                LST:    .ASCII  'LAST '
                                ADRESS: .ASCII  'MEMORY ADDRESS IS '

                                DIGITS: .ASCII  '00000000'

                                SPACE1: .ASCIZ  ' '
                                PASSMG: .ASCII  'PASS='
                                PASSNM: .ASCIZ  ' '
                                        .EVEN
                                PLACE:  .WORD   0
                                        .SBTTL  MEMORY ADDRESS TESTS

                                ;THIS TEST ADDRESS MEMORY UP TO 128k AND PROVES 'UNIQNESS' OF ALL
```

```
                                        ;MEMORY ADDRESS IN A 32K SEGMENT. THE TEST WRITES INTO EACH MEMORY
                                        ;ADDRESS THE VALUE OF THAT ADDRESS AND THEN CHECKS FOR THE CORRECT
                                        ;DATA IN EACH ADDRESS.
                                        ;THE TWELVE MOST SIGNIFICANT BITS OF THE LAST AVAILABLE MEMORY ADDRESS
                                        ;IS STORED IN R5.
                                        ;STARTING INTRUCTIONS
                                        ;       LOAD ADDRESS=200
                                        ;       PRESS START
                                        ;       STACK POINTER IS AT 500
                                        ;*************RESTART AT 162 TO RESTORE LOADER*******************
                                        ;MEMORY ADDRESS TEST
002376  012737  002440  000212  START:  MOV     #START1,@#212   ;CHANGE START ADDRESS
002404  012706  000500                  MOV     #STKPTR,SP      ;SET UP STACK PTR
002410  004767  177476                  JSR     PC,$LDR         ;GO SAVE MONITOR & LOADERS
002414  004567  176350                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
002420  007516                          RESLDR
002422  005037  000756                  CLR     @#ERCNT         ;CLEAR ERROR COUNT
002426  005037  000760                  CLR     @#LDDISP        ;CLEAR DISPLAY REGISTER STORAGE LOCN
002432  013737  000760  177570          MOV     @#LDDISP,@#DISPLAY      ;CLEAR DISPLAY REGISTER
002440  012706  000500          START1: MOV     #STKPTR,SP      ;SET STACK PTR
002444  005037  004524                  CLR     @#PEFLG         ;CLEAR PARITY ERROR INDICATORS
002450  012727  002440                  MOV     #START1,(PC)+   ;LOAD PARITY ERROR RESTART ADDRESS
002454  000000          PERSTRT:.WORD   0               ;CONTAINS RESTART ADDRESS AFTER PAR ERR
002456  005037  000752                  CLR     @#ICNT          ;CLEAR PASS COUNT
002462  005037  000764                  CLR     @#RELOCF        ;CLEAR RELOACATION FACTOR
002466  012737  000502  000024          MOV     #PDWN,@#PFVEC   ;SET POWER FAIL TRAP VECTOR
002474  005037  000026                  CLR     @#PFVEC+2

                                        ;CHECK IF MEMORY MANAGEMENT IS AVAILABLE
002500  005067  176256                  CLR     MMAVA           ;CLEAR MEM MGMT AVAILABLE INDICATOR
002504  032737  010000  177570          BIT     #BIT12,@#SWR    ;CHECK IF TO RUN WITH MEM MGMT
002512  001020                          BNE     1$              ;DO NOT USE MEM MGMT IF SW12 WAS SET
002514  012737  002554  000004          MOV     #1$,@#ERRVEC    ;SET TIME OUT TRAP
002522  005037  177572                  CLR     @#SR0           ;REFERENCE MEM MGMT
002526  005167  176230                  COM     MMAVA           ;SET INDICATOR TO -1 IF AVAILABLE
002532  012737  000020  172516          MOV     #20,@#SR3       ;SET 22 BIT MODE
002540  022737  000020  172516          CMP     #20,@#SR3       ;DID IT SET?
002546  001002                          BNE     1$              ;NO--BRANCH
002550  006367  176206                  ASL     MMAVA           ;YES--SET INDICATOR TO -2
002554  004767  002552          1$:     JSR     PC,.MAMF        ;GO ENABLE PARITY ACTION


                                        ;ROUTINE TO WRITE VALUE OF MEMORY ADDRESS INTO MEMORY ADDRESS
                                        ;FOR EXAMPLE ROUTINE WRITES 20000 INTO LOCATION 20000
002560  012737  002620  000004  WRTUP:  MOV     #DONE0,@#ERRVEC ;SET TIME OUT TRAP VECTOR
002566  010701                          MOV     PC,R1           ;LOAD TRACE REGISTER
002570  004767  002716                  JSR     PC,LDMM0
002574  012737  005614  000250          MOV     #MMABT0,@#MMVEC ;SET MEM MGMT ABORT VECTOR
002602  012702  020000                  MOV     #20000,R2       ;FIRST ADDRESS
002606  010203                          MOV     R2,R3           ;LOAD CONSTANT
002610  010322                  1$:     MOV     R3,(R2)+        ;WRITE VALUE OF ADDRESS INTO ADDRESS
002612  062703  000002                  ADD     #2,R3           ;NEXT VALUE
002616  000774                          BR      1$              ;WRITE UNTIL DONE

002620  012706  000500          DONE0:  MOV     #STKPTR,SP      ;SET STACK PTR
002624  004767  176724                  JSR     PC,$FORM0       ;GO TO FORMAT ROUTINE
```

```
941  002630  004567  176134                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
942  002634  002322                          LST
943  002636  004767  175236                  JSR     PC,CRLF
944
945                                  ;ROUTINE TO CHECK THAT VALUE OF MEMORY ADDRESS WAS WRITTEN CORRECTLY
946  002642  010701                          MOV     PC,R1           ;LOAD TRACE REGISTER
947  002644  012702  020000                  MOV     #20000,R2       ;SET R2
948  002650  012737  002724  000004          MOV     #DONE1,@#ERRVEC ;SET TIME OUT TRAP
949  002656  010200                          MOV     R2,R0
950  002660  162700  000002                  SUB     #2,R0           ;SUBTRACT 2
951  002664  004767  002622                  JSR     PC,LDMM0
952  002670  062700  000002          1$:     ADD     #2,R0
953  002674  012203                          MOV     (R2)+,R3        ;GET WRITTEN VALUE
954  002676  020003                          CMP     R0,R3           ;CHECK
955  002700  001402                          BEQ     2$
956  002702  104400                          HLT                     ;ERROR! TO DETERMINE WHICH ADDRESS WAS
957  002704  000771                          BR      1$
958  002706  005142                  2$:     COM     -(R2)
959  002710  005112                          COM     (R2)
960  002712  012203                          MOV     (R2)+,R3
961  002714  020003                          CMP     R0,R3
962  002716  001764                          BEQ     1$
963  002720  104400                          HLT
964                                  ;WRITTEN IMPROPERLY EXAMINE R2. NEXT EXAMINE MEM MGMT REGISTER KIPAR1
965                                  ;(IF MEM MGMT IS AVAILABLE). ADD R2 AND KIPAR1 TOGETHER AS SHOWN BELOW
966
967                                  ;       R2-2                    0 00X XXX XXX XXX XXX
968                                  ;       KIPAR1(772342)  Y YYY YYY YYY YYY YYY
969                                  ;       ADDRESS         Z ZZZ ZZZ ZZZ ZZZ ZZZ ZZZ ZZZ
970
971  002722  000762                          BR      1$
972  002724  012706  000500          DONE1:  MOV     #STKPTR,SP      ;SET STACK PTR
973  002730  010701                          MOV     PC,R1           ;LOAD TRACE REGISTER
974
975                                  ;ROUTINE TO WRITE 1'S COMPLEMENT VALUE OF ADDRESS INTO ADDRESS
976                                  ;FOR EXAMPLE ROUTINE WRITES 157777 INTO ADDRESS 20000
977
978  002732  005767  176024                  TST     MMAVA           ;MEMORY MAGNAGEMENT AVAILABLE?
979  002736  001420                          BEQ     3$
980  002740  013703  172342                  MOV     @#KIPAR1,R3     ;FIND LAST ADDRESS IF MEM MANAGE USED
981  002744  006303                          ASL     R3
982  002746  006303                          ASL     R3
983  002750  006303                          ASL     R3
984  002752  006303                          ASL     R3
985  002754  006303                          ASL     R3
986  002756  006303                          ASL     R3
987  002760  010246                          MOV     R2,-(SP)        ;DEVELOP COMPLEMENT OF LAST ADDRESS
988  002762  042716  020000                  BIC     #20000,(SP)     ;SAVE BITS IF MEMORY IS NOT A MULTIPLE OF 4K
989  002766  062603                          ADD     (SP)+,R3
990  002770  012737  005646  000250          MOV     #MMABT1,@#MMVEC ;SET ABORT VECTOR
991  002776  000403                          BR      2$
992  003000  162702  000002          3$:     SUB     #2,R2           ;R2=LAST ADDRESS
993  003004  010203                          MOV     R2,R3
994  003006  005103                  2$:     COM     R3              ;COMPLEMENT VALUE IN R3
995  003010  062703  000002          1$:     ADD     #2,R3
996  003014  010342                          MOV     R3,-(R2)        ;WRITE COMPLIMENT VALUE INTO ADDRESS
```

```
  997  003016  102403                          BVS     DONE3
  998  003020  020227  017776                  CMP     R2,#17776
  999  003024  001371                          BNE     1$
 1000
 1001                                 ;SET UP TO CHECK COMPLEMENT DATA WRITTEN DOWN
 1002  003026  000240                DONE3:  NOP
 1003  003030  010701                        MOV     PC,R1           ;LOAD TRACE REGISTER
 1004  003032  005767  175724                TST     MMAVA           ;CHECK IF MM IS AVAIL
 1005  003036  001406                        BEQ     1$
 1006  003040  012737  000200  172342        MOV     #200,@#KIPAR1   ;INIT KIPAR1
 1007  003046  012737  005614  000250        MOV     #MMABTO,@#MMVEC ;SET ABORT VECTOR
 1008  003054  012737  003114  000004  1$:   MOV     #DONE4,@#ERRVEC
 1009  003062  012702  020000                MOV     #20000,R2       ;FIRST ADDRESS
 1010  003066  010200                        MOV     R2,R0
 1011  003070  005100                        COM     R0              ;FIRST DATA (COM OF ADDRESS)
 1012  003072  062700  000002                ADD     #2,R0
 1013  003076  162700  000002          2$:   SUB     #2,R0
 1014  003102  012203                        MOV     (R2)+,R3        ;GET VALUE
 1015  003104  020003                        CMP     R0,R3           ;CHECK
 1016  003106  001773                        BEQ     2$
 1017  003110  104400                        HLT
 1018  003112  000771                        BR      2$
 1019  003114  000240                DONE4:  NOP
 1020
 1021                                 ;ROUTINE TO WRITE BANK # INTO ALL ADDRESSES IN A 4K BANK
 1022  003116  012737  003164  000004        MOV     #DONE4A,@#ERRVEC;SET TIME OUT TRAP VECTOR
 1023  003124  010701                        MOV     PC,R1
 1024  003126  004767  002360                JSR     PC,LDMMO
 1025  003132  012737  005614  000250        MOV     #MMABTO,@#MMVEC
 1026  003140  012702  020000                MOV     #20000,R2
 1027  003144  005000                        CLR     R0
 1028  003146  005200                  1$:   INC     R0              ;R0 WILL BE DATA WRITTEN
 1029  003150  012704  010000                MOV     #4096.,R4       ;SET 4K COUNTER
 1030  003154  010022                  2$:   MOV     R0,(R2)+        ;WRITE BANK # INTO ALL ADDRESSES
 1031  003156  005304                        DEC     R4
 1032  003160  001375                        BNE     2$
 1033  003162  000771                        BR      1$
 1034
 1035  003164  022626                DONE4A: CMP     (SP)+,(SP)+     ;ADJUST STACK PTR
 1036
 1037                                 ;CHECK THAT DATA WRITTEN ABOVE CAN BE READ
 1038  003166  012737  003234  000004        MOV     #DONE4B,@#ERRVEC
 1039  003174  010701                        MOV     PC,R1
 1040  003176  004767  002310                JSR     PC,LDMMO
 1041  003202  012702  020000                MOV     #20000,R2
 1042  003206  005000                        CLR     R0
 1043  003210  005200                  1$:   INC     R0
 1044  003212  012704  010000                MOV     #4096.,R4
 1045  003216  012203                  2$:   MOV     (R2)+,R3
 1046  003220  020003                        CMP     R0,R3
 1047  003222  001401                        BEQ     .+4
 1048  003224  104400                        HLT
 1049  003226  005304                        DEC     R4
 1050  003230  001372                        BNE     2$
 1051  003232  000766                        BR      1$
 1052  003234  022626                DONE4B: CMP     (SP)+,(SP)+
```

```
1053
1054                                        ;ROUTINE TO WRITE CONSTANT DATA INTO 4K
1055                                        ;BANK STARTING WITH LAST MEMORY LOCATION
1056  003236  010701                        MOV     PC,R1
1057  003240  012737  005646  000250        MOV     #MMABT1,@#MMVEC
1058  003246  162702  000002                SUB     #2,R2
1059  003252  005000                        CLR     R0
1060  003254  005300                1$:     DEC     R0
1061  003256  012704  010000                MOV     #4096.,R4
1062  003262  010042                2$:     MOV     R0,-(R2)
1063  003264  102406                        BVS     DONE4C
1064  003266  020227  017776                CMP     R2,#17776       ;CHECK IF DONE
1065  003272  001403                        BEQ     DONE4C
1066  003274  005304                        DEC     R4
1067  003276  001371                        BNE     2$
1068  003300  000765                        BR      1$
1069
1070  003302  012737  003414  000004  DONE4C: MOV     #DONE4D,@#ERRVEC
1071  003310  010701                        MOV     PC,R1
1072  003312  004767  002174                JSR     PC,LDMMO
1073  003316  012737  005614  000250        MOV     #MMABT0,@#MMVEC ;SET ABORT VECTOR
1074  003324  012702  020000                MOV     #20000,R2
1075  003330  022704  010000        1$:     CMP     #4096.,R4       ;CHECK IF WRITE ABOVE STARTED ON
1076                                                                ;4K BOUNDARY
1077  003334  001415                        BEQ     2$
1078  003336  012203                        MOV     (R2)+,R3
1079  003340  020003                        CMP     R0,R3
1080  003342  001402                        BEQ     4$
1081  003344  104400                        HLT
1082  003346  000406                        BR      5$
1083  003350  005142                4$:     COM     -(R2)
1084  003352  005112                        COM     (R2)
1085  003354  012203                        MOV     (R2)+,R3
1086  003356  020003                        CMP     R0,R3
1087  003360  001401                        BEQ     5$
1088  003362  104400                        HLT
1089  003364  005204                5$:     INC     R4
1090  003366  001360                        BNE     1$
1091  003370  005200                2$:     INC     R0
1092  003372  012704  010000                MOV     #4096.,R4
1093  003376  012203                3$:     MOV     (R2)+,R3
1094  003400  020003                        CMP     R0,R3
1095  003402  001401                        BEQ     .+4
1096  003404  104400                        HLT
1097  003406  005304                        DEC     R4
1098  003410  001372                        BNE     3$
1099  003412  000766                        BR      2$
1100
1101  003414  022626                DONE4D: CMP     (SP)+,(SP)+
1102  003416  005737  000042                TST     @#42            ;BRANCH IF PROGRAM WAS NOT
1103  003422  001406                        BEQ     BEGIN1          ;LOADED VIA ACT11 IN QV OR AA MODES
1104  003424  005767  000610                TST     LOGICAL+2       ;BRANCH IF NOT IN QV MODE
1105  003430  100003                        BPL     BEGIN1
1106  003432  012737  000001  004170        MOV     #1,@#ENDCT      ;SET ENDCT TO DO 1 PASS ONLY IN QV
1107                                        .SBTTL  WORST CASE NOISE TESTS
1108                                ;THIS TEST WRITES MEMORY WORST CASE NOISE TEST PATTERNS THROUGHOUT
```

```
1109                                        ;MEMORY AND CHECKS THAT THEY CAN BE WRITTEN AND READ.
1110                                        ;SET UP TRAP VECTORS
1111  003440  012706  000500          BEGIN1: MOV     #STKPTR,SP      ;SET STACK PTR
1112  003444  004767  001662                  JSR     PC,,MAMF        ;GO ENABLE PARITY ACTION
1113  003450  004767  003740                  JSR     PC,CKSWR        ;GO CHECK SWITCHES
1114  003454  005027                          CLR     (PC)+           ;SET INDICATOR TO WRITE NORMAL 3X9 PAT
1115  003456  000000                  PARPAT: .WORD   0
1116  003460  022767  177776  175274        CMP     #-2,MMAVA       ;22 BIT MODE
1117  003466  001002                        BNE     DONE6           ;NO--BRANCH
1118  003470  004767  001770                JSR     PC,MARGIN       ;YES--GO SETUP MARGINS
1119
1120
1121                                        ;WRITE 3 XOR 9 TEST PATTERN STARTING AT ADDRESS 20000
1122                                        ;NOTE PATTERN IS NORMAL 3 XOR 9 IF NO PARITY MEMORY IS AVAILABLE,
1123                                        ;AND IS A MODIFIED PATTERN IF PARITY MEMORY IS AVAILABLE.
1124                                        ;THE CONTENTS OF PARPAT IF 0/NOT 0 INDICATE IF NORMAL/MODIFIED PATTERN
1125                                        ;IS BEING USED IN TESTS BELOW.
1126  003474  012706  000500          DONE6:  MOV     #STKPTR,SP      ;SET STACK PTR
1127  003500  010701                        MOV     PC,R1           ;UPDATE TRACE REGISTER
1128  003502  012737  003522  000004        MOV     #DONE7,@#ERRVEC ;SET TIME OUT TRAP VECTOR
1129  003510  012746  000001                MOV     #1,-(SP)        ;PUSH STARTING BANK # ON STACK
1130  003514  005046                        CLR     -(SP)           ;PUSH # OF 256. WORD BLOCKS TO WRITE
1131  003516  004767  002340                JSR     PC,,3X9         ;CALL ROUTINE TO WRITE 3XOR9 PATTERN
1132
1133                                        ;CHECK 3 XOR 9 TEST PATTERN WRITTEN ABOVE
1134  003522  012737  001126  000004  DONE7:  MOV     #ERRTRP,@#ERRVEC
1135  003530  016600  000006                MOV     6(SP),R0        ;GET # OF 256. WORD BLOCKS WRITTEN
1136  003534  005400                        NEG     R0              ;FORM TWO'S COMPLEMENT
1137  003536  010027                        MOV     R0,(PC)+        ;SAVE # OF 256 WORD BLOCKS
1138  003540  000000                  WDS.256:.WORD   0               ;CONTAINS # OF 256 WORD BLOCKS IN MEM.
1139  003542  012706  000500                MOV     #STKPTR,SP      ;SET STACK PTR
1140  003546  010701                        MOV     PC,R1           ;SET SCOPE PTR
1141  003550  012746  000001                MOV     #1,-(SP)        ;PUSH BANK # ON THE STACK
1142  003554  010046                        MOV     R0,-(SP)        ;PUSH # OF 256. WORD BLOCKS TO WRITE
1143  003556  004767  002520                JSR     PC,,,3X9        ;GO CHECK DATA WRITTEN
1144
1145                                        ;SETUP TO RUN MODIFIED 3 XOR 9 PATTERN IF PARITY MEMORY IS AVAILABLE
1146  003562  022767  177776  175172        CMP     #-2,MMAVA
1147  003570  001403                        BEQ     1$
1148  003572  005737  005402                TST     @#PARAVA        ;BRANCH IF PARITY MEMORY IS NOT AVAIL
1149  003576  001406                        BEQ     DONE8
1150  003600  005737  003456          1$:   TST     @#PARPAT        ;BRANCH IF PARITY PAT JUST WRITTEN
1151  003604  001003                        BNE     DONE8
1152  003606  010637  003456                MOV     SP,@#PARPAT     ;SET INDICATOR TO WRITE 3X9 PAR PAT
1153  003612  000730                        BR      DONE6           ;REPEAT TEST USING MODIFIED 3X9 PATTERN
1154
1155                                        ;WRITE 8 XOR 13 TEST PATTERN STARTING AT ADDRESS 40000
1156  003614  012706  000500          DONE8:  MOV     #STKPTR,SP      ;SET STACK PTR
1157  003620  012737  003642  000004        MOV     #DONE9,@#ERRVEC ;SET TIME OUT TRAP VECTOR
1158  003626  010701                        MOV     PC,R1           ;UPDATE TRACE REGISTER
1159  003630  012746  000002                MOV     #2,-(SP)        ;PUSH STARTING BANK # ON THE STACK
1160  003634  005046                        CLR     -(SP)           ;PUSH # OF BANKS TO WRITE ON THE STACK
1161  003636  004767  003234                JSR     PC,,8X13        ;GO TO ROUTINE TO WRITE DATA
1162
1163                                        ;CHECK 8 XOR 13 TEST PATTERN WRITTEN ABOVE
1164  003642  012706  000500          DONE9:  MOV     #STKPTR,SP      ;SET STACK PTR
```

```
                                                .SBTTL  RANDOM DATA,ROTATING 1/0 TESTS
                                        ;RANDOM DATA TEST. THIS TEST MOVES THE PROGRAM CODE THROUGHOUT MEMORY
003646 010701                           RANTST: MOV     PC,R1                   ;SET TRACE POINTER
003650 012737 001126 000004                     MOV     #7$,@#ERRVEC            ;SET TIME OUT TRAP
003656 012746 000002                            MOV     #2,-(SP)
003662 005404                                   NEG     R4
003664 042704 000001                            BIC     #1,R4                   ;SET 4K BANK COUNT TO 8K INCREMENT
003670 001403                                   BEQ     DONE10                  ;DO NOT CHECK IF ONLY 12K
003672 010446                                   MOV     R4,-(SP)
003674 004767 003204                            JSR     PC,...8X13              ;GO CHECK 8 XOR 13 PATTERN WRITTEN ABOVE

003700 000005                           DONE10: RESET                           ;DISABLE MEM MGMT AND PARITY ACTION

003702 010701                                   MOV     PC,R1                   ;SET TRACE POINTER
003704 012737 004042 000004                     MOV     #7$,@#ERRVEC            ;SET TIME OUT TRAP
003712 005767 175044                            TST     MMAVA                   ;CHECK IF MEM MGMT IS AVAILABLE
003716 001412                                   BEQ     1$                      ;BRANCH IF NOT AVAILABLE
003720 004767 001566                            JSR     PC,LDMM0                ;GO SET UP MEM MGMT
003724 105237 172301                            INCB    @#KIPDR0+1              ;ALLOW 4K ADDRESSING IN FIRST 4K
003730 012737 077406 172304                     MOV     #200*256.-400+UP+RW,@#KIPDR2    ;SET KIPDR2=RW UP 200 BLOCKS
003736 012737 000400 172344                     MOV     #400,@#KIPAR2
003744 012702 020000                    1$:     MOV     #20000,R2               ;SET 'TO' ADDRESS POINTER
003750 005004                                   CLR     R4                      ;SET 'FROM' ADDRESS POINTER
003752 012705 004000                    2$:     MOV     #2048.,R5               ;SET 4K WORD COUNT
003756 012422                           3$:     MOV     (R4)+,(R2)+             ;MOVE CODE
003760 012422                                   MOV     (R4)+,(R2)+
003762 005305                                   DEC     R5                      ;DECREMENT 4K WORD COUNTER
003764 001374                                   BNE     3$

003766 012705 005405                            MOV     #4096.-PLACE+1,R5       ;SET 4K WORD COUNTER
003772 014400                           4$:     MOV     -(R4),R0                ;GET 'GOOD' DATA
003774 014203                                   MOV     -(R2),R3                ;GET 'BAD' DATA
003776 020003                                   CMP     R0,R3                   ;COMPARE 'GOOD' & 'BAD' DATA
004000 001403                                   BEQ     5$
004002 005722                                   TST     (R2)+                   ;STEP ADDRESS FOR ERROR ROUTINE
004004 104400                                   HLT                             ;REPORT ERROR
004006 005742                                   TST     -(R2)                   ;RESTORE ADDRESS POINTER
004010 005305                           5$:     DEC     R5                      ;DECREMENT 4K WORD COUNTER
004012 001367                                   BNE     4$                      ;LOOP UNTIL 4K WORDS CHECKED

004014 005767 174742                            TST     MMAVA                   ;CHECK IF MEM MGMT IS AVAILABLE
004020 001405                                   BEQ     6$                      ;BRANCH IF NOT AVAILABLE
004022 005237 172342                            INC     @#KIPAR1
004026 005237 172344                            INC     @#KIPAR2
004032 000744                                   BR      1$
004034 062702 000100                    6$:     ADD     #64.,R2                 ;STEP ADDRESS
004040 000744                                   BR      2$
004042 012706 000500                    7$:     MOV     #STKPTR,SP              ;RESET STACK PTR
004046 012737 001126 000004                     MOV     #ERRTRP,@#ERRVEC        ;RESTORE ERROR TRAP VECTOR

                                        ;ROTATING 0 TEST. THIS TEST ROTATES A SINGLE '0' THROUGH MEMORY
004054 012767 177777 003276             ROTO:   MOV     #-1,.CONST              ;SET CONSTANT =177777
004062 012746 000001                            MOV     #1,-(SP)                ;SET BANK #1
004066 016746 177446                            MOV     WD$.256,-(SP)           ;GET # OF 256. WORD BLOCKS IN MEMORY
```

```
1221  004072  004767  003244                  JSR     PC,WRTPAT       ;GO WRITE 1'S THROUGHOUT MEMORY
1222  004076  010701                          MOV     PC,R1           ;SET SCOPE PTR
1223  004100  012746  000001                  MOV     #1,-(SP)        ;SET STARTING BANK #
1224  004104  016746  177430                  MOV     WDS.256,-(SP)   ;SET # OF 256. WORD BLOCKS TO CHECK
1225  004110  004767  002776                  JSR     PC,.ROT0        ;GO TO ROTATE 0 ROUTINE
1226
1227                                  ;ROTATING 1 TEST THIS TEST ROTATES A SINGLE '1' BIT THROUGH ALL OF
1228                                  ;MEMORY
1229  004114  005067  003240          ROT1:   CLR     .CONST          ;CLEAR CONSTANT
1230  004120  012746  000001                  MOV     #1,-(SP)        ;PUSH STARTING BANK ONTO STACK
1231  004124  016746  177410                  MOV     WDS.256,-(SP)   ;AND # OF 256. WORD BLOCKS IN MEMORY
1232  004130  004767  003206                  JSR     PC,WRTPAT       ;GO WRITE 0'S THROUGHOUT MEMORY
1233  004134  010701                          MOV     PC,R1           ;SET SCOPE PTR
1234  004136  012746  000001                  MOV     #1,-(SP)        ;SET STARTING BANK #
1235  004142  016746  177372                  MOV     WDS.256,-(SP)   ;SET # OF 256. WORD BLOCKS TO CHECK
1236  004146  004767  003034                  JSR     PC,.ROT1        ;GO ROTATE A '1' BIT THROUGHOUT MEMORY
1237
1238                                  ;END  OF CYCLE
1239  004152  000005                  END:    RESET
1240  004154  010701                          MOV     PC,R1           ;UPDATE TRACE REGISTER
1241  004156  012706  000500                  MOV     #STKPTR,SP      ;SET STACK PTR
1242  004162  005237  000752                  INC     @#ICNT          ;INCREMENT PASS COUNT
1243  004166  022737                          CMP     (PC)+,@(PC)+    ;CHECK FOR LAST PASS
1244  004170  000006                  ENDCT:  .WORD   6               ;MAKE 5 PASSES
1245  004172  000752                          .WORD   ICNT            ;PASS COUNT ADDRESS
1246  004174  001405                          BEQ     DONE            ;BRANCH IF LAST PASS COMPLETED
1247  004176  004567  174566                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1248  004202  010104                          ASTERISK
1249  004204  000137  003440                  JMP     @#BEGIN1
1250  004210                          DONE:
1251  004210  004567  174554                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1252  004214  010106                          ENDMSG
1253  004216  105737  177564                  TSTB    @#TPS           ;WAIT FOR BELL TO RING
1254  004222  100375                          BPL     .-4
1255  004224  013700  000042                  MOV     @#42,R0         ;GET DECTAPE MONITOR RETURN ADDRESS
1256  004230  001406                          BEQ     FINISH
1257  004232  004767  175752                  JSR     PC,$RLDR        ;RESTORE MONITOR & LOADERS
1258  004236  004710                  LOGICAL:JSR     PC,(R0)         ;GO TO DECTAPE MONITOR
1259  004240  000240                          NOP
1260  004242  000240                          NOP
1261  004244  000240                          NOP
1262  004246  000167  176156          FINISH: JMP     START1
1263
1264                                          .SBTTL  PROGRAM SUBROUTINES
1265                                          .SBTTL  RELOCATION ROUTINES
1266                                  ;ROUTINE TO RELOCATE PROGRAM CODE
1267  004252  012500                  RELOC:  MOV     (R5)+,R0        ;GET FROM ADDRESS
1268  004254  011502                          MOV     (R5),R2         ;GET TO ADDRESS
1269  004256  010203                          MOV     R2,R3
1270  004260  062703  017776                  ADD     #17776,R3       ;MOVES 4K
1271  004264  012737  004334  000004          MOV     #4$,@#ERRVEC    ;SET TIME OUT TRAP
1272  004272  005004                          CLR     R4              ;CLEAR RELOCATION SUCCESSFUL INDICATOR
1273  004274  005723                          TST     (R3)+           ;CHECK IF MEMORY IS AVAILABLE
1274  004276  012022                  1$:     MOV     (R0)+,(R2)+     ;RELOCATE
1275  004300  020203                          CMP     R2,R3           ;RELOCATION COMPLETE?
1276  004302  001375                          BNE     1$
```

```
1277  004304  011503                        MOV     (R5),R3
1278  004306  020203                2$:     CMP     R2,R3
1279  004310  001413                        BEQ     5$                      ;BRANCH IF DONE
1280  004312  024042                        CMP     -(R0),-(R2)             ;CHECK THAT DATA WAS RELOCATED PROPERLY
1281  004314  001774                        BEQ     2$
1282  004316  005703                        TST     R3                      ;CHECK IF RELOCATING BACK TO 000000
1283  004320  001403                        BEQ     3$
1284  004322  104400                        HLT                             ;ERROR! CANNOT RELOCATE PROGRAM CODE
1285                                                                        ;TO UPPER MEMORY BANK PROPERLY
1286  004324  000000                        HALT
1287  004326  000767                        BR      2$                      ;CONTINUE RELOCATING AT YOUR PERIL
1288  004330  000000                3$:     HALT                            ;ERROR! CANNOT RELOCATE CODE BACK TO
1289                                                                        ;TO 000000 PROPERLY
1290  004332  000777                        BR
1291  004334  022626                4$:     CMP     (SP)+,(SP)+             ;RESTORE STACK PTR
1292  004336  005104                        COM     R4
1293  004340  000240                5$:     NOP
1294  004342  012702  000764               MOV     #RELOCF,R2              ;GET ADDRESS OF RELOCATION FACTOR
1295  004346  061502                        ADD     (R5),R2                 ;ADD FACTOR
1296  004350  012512                        MOV     (R5)+,(R2)              ;RELOCATED RELOCF NOW CONTAINS RELOCATION
1297                                                                        ;FACTOR
1298  004352  000205                        RTS     5                       ;RETURN, R4=-1 IF NO RELOCATION
1299
1300
1301                                  ;ROUTINE TO RELOCATE PROGRAM CODE FROM ORIGINAL POSITION (0-4K) TO
1302                                  ;TOP OF MEMORY.
1303  004354  012700  020000       RELOCP: MOV     #20000,R0               ;SET UP TO SCAN FOR TOP OF MEMORY
1304  004360  012737  000006  000004        MOV     #ERRVEC+2,@#ERRVEC
1305  004366  062700  020000        1$:     ADD     #20000,R0               ;INCREMENT SCAN ADDRESS
1306  004372  000261                        SEC                             ;SET TIME OUT INDICATOR
1307  004374  005710                        TST     (R0)                    ;CHECK FOR EXISTANT MEMORY
1308  004376  103373                        BCC     1$                      ;'C' WILL BE CLEAR IF MEMORY EXISTS
1309  004400  012737  001126  000004        MOV     #ERRTRP,@#ERRVEC
1310  004406  162700  020000                SUB     #20000,R0               ;ADJUST TO LAST EXISTANT 4K
1311  004412  010067  000006                MOV     R0,2$                   ;PASS RELOCATION ADDRESS TO RELOC ROUTINE
1312  004416  004567  177630                JSR     R5,RELOC                ;RELOCATE PROGRAM
1313  004422  000000                        000000                          ;FROM ADDRESS 000000
1314  004424  000000                2$:     .WORD   0                       ;TO LAST 4K BANK
1315  004426  004567  174336                JSR     R5,$PRINT               ;GO TO PRINT ROUTINE
1316  004432  010045                        RELOCM
1317  004434  016746  177764                MOV     2$,-(SP)                ;PASS TO O2A ROUTINE
1318  004440  062716  010122                ADD     #REL24K,(SP)            ;SET UP RESTART ADDRESS
1319  004444  004767  175344                JSR     PC,O2A                  ;TYPE RESTART ADDRESS
1320  004450  011667  000006                MOV     (SP),3$                 ;SAVE RETURN ADDRESS IN 3$ BELOW
1321  004454  066706  177744                ADD     2$,SP                   ;RESET STACK PTR
1322  004460  012716                        MOV     (PC)+,(SP)              ;GET RETURN ADDRESS
1323  004462  000000                3$:     .WORD   0                       ;CONTAINS RETURN PC
1324  004464  066716  177734                ADD     2$,(SP)                 ;ADJUST RETURN PC
1325  004470  000207                        RTS     PC
1326
1327                                          .SBTTL  MA/MF PARITY ERROR SERVICE ROUTINE
1328                                  ;WHEN MA/MF A PARITY ERROR IS DETECTED THIS ROUTINE SCANS MEMORY FOR THE
1329                                  ;ADDRESS CAUSING THE PARITY ERROR. WHEN THE ADDRESS IS LOCATED THE ROUTINE
1330                                  ;HALTS WITH THE ADDRESS+2 IN R0. TO CONTINUE AFTER THE ERROR PRESS CONTINUE.
1331  004472  010067  000170       .PARSRV:MOV     R0,SAVR0                ;SAVE R0 IN SAVR0
1332  004476  012700  004670                MOV     #SAVR0+2,R0
```

```
1333  004502  010120                        MOV     R1,(R0)+
1334  004504  010220                        MOV     R2,(R0)+
1335  004506  010320                        MOV     R3,(R0)+
1336  004510  010420                        MOV     R4,(R0)+
1337  004512  010520                        MOV     R5,(R0)+
1338  004514  004567  174250                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1339  004520  004702                        PARERR
1340  004522  005027                        CLR     (PC)+           ;CLEAR PARITY ERROR INDICATORS
1341  004524     000          PEFLG:  .BYTE   0               ;NOT 0/0 =PAR ERR/NO PAR ERR
1342  004525     000          PENFLG: .BYTE   0               ;NOT 0/0=PAR ERR DETECTED/NOT DETECTED ON SCAN
1343  004526  012737  004574  000114        MOV     #2$,@#PARVEC            ;SET PARITY ERROR TRAP
1344  004534  012737  004632  000004        MOV     #4$,@#ERRVEC            ;SET TIME OUT TRAP VECTOR
1345  004542  005002                        CLR     R2
1346  004544  005767  174212                TST     MMAVA           ;CHECK IF MEM MGMT IS AVAILABLE
1347  004550  001407                        BEQ     1$              ;BRANCH IF NOT AVAILABLE
1348  004552  004767  000734                JSR     PC,LDMM0        ;SET UP MEM MGMT
1349  004556  105237  172301                INCB    @#KIPDR0+1      ;ALLOW FULL 4K PAGE ADDRESSING
1350  004562  012737  005614  000250        MOV     #MMABT0,@#MMVEC ;SET MEM MGMT ABORT TRAP VECTOR
1351  004570  012200          1$:     MOV     (R2)+,R0        ;SCAN ALL ADDRESSES
1352  004572  000776                        BR      1$
1353  004574  110667  177724          2$:     MOVB    SP,PEFLG        ;SET PARITY ERROR FOUND INDICATOR
1354  004600  010003                        MOV     R0,R3
1355  004602  104400                        HLT                     ;PARITY ERROR! ADDRESS+2 IS IN R2
1356  004604  000002                        RTI                     ;CONTINUE SCAN
1357  004606  000240          3$:     NOP                     ;INSERT HALT INST TO EXAMINE PARITY REGS
1358  004610  005067  177710                CLR     PEFLG           ;CLEAR PARITY ERROR INDICATORS
1359  004614  012706  000500                MOV     #STKPTR,SP      ;RESET STACK PTR
1360  004620  000005                        RESET
1361  004622  004767  000504                JSR     PC,.MAMF        ;GO ENABLE PARITY ERROR DETECTION
1362  004626  000177  175622                JMP     @PERSTRT        ;RESTART SELECTED PROGRAM
1363
1364                                  ;SERVICE ROUTINE IF PARITY ERROR NOT DETECTED ON SCAN
1365  004632  105767  177666          4$:     TSTB    PEFLG           ;BRANCH IF PARITY ERROR WAS
1366  004636  001363                        BNE     3$              ;DETECTED ON SCAN
1367  004640  016602  000004                MOV     4(SP),R2        ;GET PC AT TIME OF ERROR
1368  004644  162702  000002                SUB     #2,R2           ;BACK IT UP
1369  004650  110667  177651                MOVB    SP,PENFLG       ;SET IND = NO PAR ERROR DETECTED ON SCAN
1370  004654  004567  174110                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1371  004660  004723                        NOFIND
1372  004662  104400                        HLT                     ;ERROR! PARITY ERROR NOT DETECTED ON SCAN
1373  004664  000750                        BR      3$
1374                                  ;THE BELOW 6 WORDS CONTAINS THE SAVED CONTENTS OF R0-R5 WHEN THE
1375                                  ;PARITY ERROR OCCURRED
1376  004666  000000          SAVR0:  .WORD   0
1377  004670  000000          SAVR1:  .WORD   0
1378  004672  000000          SAVR2:  .WORD   0
1379  004674  000000          SAVR3:  .WORD   0
1380  004676  000000          SAVR4:  .WORD   0
1381  004700  000000          SAVR5:  .WORD   0
1382
1383  004702  005015  040520  044522  PARERR: .ASCIZ  <15><12>'PARITY ERROR'<15><12>
1384  004710  054524  042440  051122
1385  004716  051117  005015     000
1386  004723     116  052117  043040  NOFIND: .ASCIZ  'NOT FOUND ON SCAN'<15><12>
1387  004730  052517  042116  047440
1388  004736  020116  041523  047101
```

```
1389  004744  005015      000
1390          004750                          .EVEN
1391
1392
1393          177740                  MEMLO=177740
1394          177742                  MEMHI=177742
1395          177744                  MEMERR=177744
1396
1397
1398  004750  005767  177550          .22PAR: TST     PEFLG           ;BEEN HERE BEFORE
1399  004754  001403                          BEQ     1$              ;BRANCH IF NO
1400  004756  000000                          HALT                    ;YES -- DOUBLE PARITY ERROR
1401  004760  000177  175470                  JMP     @PERSTRT
1402  004764  010667  177534          1$:     MOV     SP,PEFLG        ;SET PARITY ERROR FLAG
1403  004770  005737  177570                  TST     @#SWR           ;HALT ON ERROR?
1404  004774  100001                          BPL     100$            ;BRANCH IF NO
1405  004776  000000                          HALT                    ;YES
1406  005000  013746  177744          100$:   MOV     @#MEMERR,-(SP)  ;SAVE MEMORY ERROR REG
1407  005004  013701  177740                  MOV     @#MEMLO,R1      ;GET ADDRESS OF WHERE THE PARITY
1408  005010  013702  177742                  MOV     @#MEMHI,R2      ;ERROR OCCURRED
1409  005014  011637  177744                  MOV     (SP),@#MEMERR   ;CLEAR THE ERROR REG
1410  005020  032737  020000  177570          BIT     #BIT13,@#SWR    ;INHIBIT ERROR TYPEOUT
1411  005026  001066                          BNE     101$            ;BRANCH IF YES
1412                                  ;PRINT  "PARITY ERROR"
1413  005030  004567  173734                  JSR     R5,$PRINT
1414  005034  004702                          PARERR
1415                                  ;PRINT  "PC=XXXXXX"
1416  005036  004567  173726                  JSR     R5,$PRINT
1417  005042  001475                          ERRPC
1418  005044  016646  000002                  MOV     2(SP),-(SP)     ;GET PC AT TIME OF PARITY ERROR
1419  005050  066716  173710                  ADD     RELOCF,(SP)
1420  005054  004767  174734                  JSR     PC,02A
1421  005060  004567  173704                  JSR     R5,$PRINT
1422  005064  002361                          SPACE1
1423                                  ;CHANGE 22-BIT ADDRESS TO OCTAL-ACSII
1424  005066  012700  002351                  MOV     #DIGITS,R0
1425  005072  012704  000010                  MOV     #8.,R4
1426  005076  012705  000003          2$:     MOV     #3,R5
1427  005102  005003                  3$:     CLR     R3
1428  005104  006301                  4$:     ASL     R1
1429  005106  106102                          ROLB    R2
1430  005110  006103                          ROL     R3
1431  005112  077504                          SOB     R5,4$
1432  005114  116320  002312                  MOVB    DIGTAB(R3),(R0)+
1433  005120  077412                          SOB     R4,2$
1434                                  ;PRINT  "MEMORY ADDRESS IS AAAAAAAA"
1435  005122  004567  173642                  JSR     R5,$PRINT
1436  005126  002327                          ADRESS
1437                                  ;PRINT  "PARITY ERROR REG=XXXXXX"
1438  005130  004567  173634                  JSR     R5,$PRINT
1439  005134  001531                          PARREG
1440  005136  011605                          MOV     (SP),R5
1441  005140  004767  174650                  JSR     PC,02A
1442  005144  004767  175211                  JSR     PC,SPACE1
1443                                  ;PRINT THE MARGIN SETTING
1444  005150  016700  173576                  MOV     ICNT,R0
```

```
1445  005154  116000  005504                        MOVB    MRGNTB(R0),R0
1446  005160  062700  005210                        ADD     #MARTBL,R0
1447  005164  010067  000004                        MOV     R0,5$
1448  005170  004567  173574                        JSR     R5,$PRINT
1449  005174  005210                        5$:     MARTBL
1450  005176  004567  173566                        JSR     R5,$PRINT
1451  005202  005317                                MARMSG
1452  005204  000177  175244                101$:   JMP     @PERSTRT
1453                                        ;MARGIN MESSAGE TABLE
1454  005210  005226                        MARTBL: NORMAL
1455  005212  000000                                0
1456  005214  005235                                ESTRB
1457  005216  005252                                LSTRB
1458  005220  005266                                LCRNT
1459  005222  005302                                HCRNT
1460  005224  005226                                NORMAL
1461
1462                                        ;MARGIN MESSAGES
1463  005226  047516  046522  046101  NORMAL: .ASCIZ  'NORMAL'
1464  005234     000
1465  005235     105  051101  054514  ESTRB:  .ASCIZ  'EARLY STROBE'
1466  005242  051440  051124  041117
1467  005250  000105
1468  005252  040514  042524  051440  LSTRB:  .ASCIZ  'LATE STROBE'
1469  005260  051124  041117  000105
1470  005266  047514  020127  052503  LCRNT:  .ASCIZ  'LOW CURRENT'
1471  005274  051122  047105  000124
1472  005302  044510  044107  041440  HCRNT:  .ASCIZ  'HIGH CURRENT'
1473  005310  051125  042522  052116
1474  005316     000
1475  005317     040  040515  043522  MARMSG: .ASCIZ  ' MARGIN'<12><15>
1476  005324  047111  006412     000
1477          005332                    .EVEN
1478
1479                                        ;ROUTINE TO ENABLE PARITY ERROR ACTION ON MA/MF PARITY MEMORIES
1480          172100                            PARCSR=172100             ;ADDRESS OF FIRST PARITY REGISTER
1481          000114                            PARVEC=114                ;PARITY ERROR INTERRUPT VECTOR ADDRESS
1482
1483  005332  032737  000040  177570  .MAMF:  BIT     #40,@#SWR          ;CHECK IF PARITY ERROR DETECTION IS TO
1484  005340  001033                          BNE     DISPAR             ;BE ENABLED. BRANCH IF NOT TO BE ENABLED
1485  005342  013746  000004                  MOV     @#ERRVEC,-(SP)     ;SAVE ERROR TRAP VECTOR
1486  005346  012737  000006  000004          MOV     #ERRVEC+2,@#ERRVEC      ;SET TIME OUT TRAP TO RETURN (VIA RTI)
1487  005354  012737  004472  000114          MOV     #.PARSRV,@#PARVEC       ;SET PARITY ERROR TRAP VECTOR
1488  005362  012737  000340  000116          MOV     #340,@#PARVEC+2    ;PRIORITY LEVEL 7 ON TRAP
1489  005370  012700  172100                  MOV     #PARCSR,R0         ;GET FIRST ADDRESS OF PARITY REGISTER
1490  005374  012702  000001                  MOV     #1,R2
1491  005400  005027                          CLR     (PC)+              ;CLEAR AVAILABILITY INDICATOR
1492  005402  000000                  PARAVA: .WORD   0                  ;CONTAINS AVAILABILITY INDICATOR
1493
1494                                        ;ENABLE ALL AVAILABLE PARITY REGISTERS
1495  005404  000262                  1$:     SEV                        ;SET TIME OUT INDICATOR
1496  005406  012720  000001                  MOV     #1,(R0)+           ;SET ACTION ENABLE IF AVAILABLE
1497  005412  102402                          BVS     2$                 ;BRANCH IF NO PARITY AVAILABLE
1498  005414  050267  177762                  BIS     R2,PARAVA          ;SET AVAILABLITY INDICATOR
1499  005420  006302                  2$:     ASL     R2                 ;SHIFT INDICATOR
1500  005422  103370                          BCC     1$
```

```
1501  005424  012637  000004                  MOV     (SP)+,@#ERRVEC  ;RESTORE ERROR TRAP VECTOR
1502  005430  022767  177776  173324  DISPAR: CMP     #-2,MMAVA
1503  005436  001011                          BNE     1$
1504  005440  012767  000001  177734          MOV     #1,PARAVA
1505  005446  012737  004750  000114          MOV     #.22PAR,@#PARVEC
1506  005454  012737  000340  000116          MOV     #340,@#PARVEC+2
1507  005462  000207                  1$:     RTS     PC              ;RETURN
1508
1509                                          .SBTTL MARGIN ROUTINE
1510                                  ;ROUTINE TO SET THE MARGINS
1511          177746                  CNTRL=177746
1512          177750                  MAINTRG=177750
1513
1514  005464  016700  173262          MARGIN: MOV     ICNT,R0         ;PASS COUNT
1515  005470  005002                          CLR     R2              ;FAST COUNTER
1516  005472  116037  005504  177750          MOVB    MRGNTB(R0),@#MAINTRG ;LOAD MAINTENANCE REG.
1517  005500  077201                  1$:     SOB     R2,1$
1518  005502  000207                          RTS     PC
1519
1520  005504      000                 MRGNTB: .BYTE   0               ;NORMAL
1521  005505      004                         .BYTE   4               ;EARLY STROBE
1522  005506      006                         .BYTE   6               ;LATE STROBE
1523  005507      010                         .BYTE   10              ;LOW CURRENT
1524  005510      012                         .BYTE   12              ;HIGH CURRENT
1525  005511      000                         .BYTE   0               ;NORMAL
1526
1527
1528                                          .SBTTL  MEM MGMT ROUTINES
1529                                  ;ROUTINE TO INITIALIZE MEMORY MANAGEMENT REGISTERS
1530  005512  000240                  LDMMO:  NOP
1531  005514  005767  173242                  TST     MMAVA
1532  005520  001434                          BEQ     1$
1533  005522  012737  000020  172516          MOV     #20,@#SR3       ;22 BIT MODE
1534  005530  012737  077006  172300          MOV     #177*256.-400+UP+RW,@#KIPDR0    ;SET KIPDR0=RW UP 177 BLOCKS
1535  005536  012737  077406  172302          MOV     #200*256.-400+UP+RW,@#KIPDR1    ;SET KIPDR1=RW UP 200 BLOCKS
1536  005544  005037  172304                  CLR     @#KIPDR2
1537  005550  005037  172344                  CLR     @#KIPAR2
1538  005554  012737  077406  172316          MOV     #200*256.-400+UP+RW,@#KIPDR7    ;SET KIPDR7=RW UP 200 BLOCKS
1539  005562  005037  172340                  CLR     @#KIPAR0
1540  005566  012737  000200  172342          MOV     #200,@#KIPAR1
1541  005574  012737  177600  172356          MOV     #177600,@#KIPAR7
1542  005602  012737  000001  177572          MOV     #1,@#SR0        ;ENABLE MEM MGMT
1543  005610  000240                          NOP
1544  005612  000207                  1$:     RTS     PC
1545
1546                                  ;MEMORY MANAGEMENT ABORT ROUTINE FOR WRITE UP
1547  005614  012702  020000          MMABT0: MOV     #20000,R2       ;RESET R2
1548  005620  062737  000200  172342          ADD     #200,@#KIPAR1   ;ADVANCE TO NEXT 4K
1549  005626  013716  177576                  MOV     @#SR2,(SP)      ;RETURN TO INSTRUCTION THAT
1550  005632  005037  177572                  CLR     @#SR0           ;DISABLE MEM MGMT
1551  005636  012737  000001  177572          MOV     #1,@#SR0        ;ENABLE MEM MGMT
1552  005644  000002                          RTI                     ;CAUSED THE ABORT
1553
1554                                  ;MEM MGMT ABORT SERVICE FOR WRITE DOWN
1555  005646  012702  040000          MMABT1: MOV     #40000,R2       ;RESET R2
1556  005652  162737  000200  172342          SUB     #200,@#KIPAR1
```

```
1557  005660  001406                          BEQ     2$
1558  005662  013716  177576                  MOV     @#SR2,(SP)
1559  005666  012737  000001  177572          MOV     #1,@#SR0        ;ENABLE MEM MGMT
1560  005674  000002                          RTI
1561  005676                          2$:
1562  005676  005037  177572                  CLR     @#SR0           ;DISABLE MEM MGMT
1563  005702  052766  000002  000002          BIS     #V,2(SP)
1564  005710  000002                          RTI
1565
1566                          ;ROUTINE TO SET UP MEMORY MANAGEMENT FOR PATTERN TESTS
1567  005712  005702                  STMM2:  TST     R2              ;CHECK IF TESTING BANK # 0
1568  005714  001442                          BEQ     2$              ;EXIT IF BANK # 0
1569  005716  005767  173040                  TST     MMAVA
1570  005722  001005                          BNE     1$              ;BRANCH IF MEM MGMT AVAILABLE
1571  005724  006002                          ROR     R2              ;ADJUST ADDRESS
1572  005726  006002                          ROR     R2
1573  005730  006002                          ROR     R2
1574  005732  006002                          ROR     R2
1575  005734  000207                          RTS     PC              ;RETURN
1576
1577  005736  004767  177550          1$:     JSR     PC,LDMM0        ;GO MAKE INITIAL SET UP
1578  005742  000302                          SWAB    R2
1579  005744  006002                          ROR     R2
1580  005746  010237  172344                  MOV     R2,@#KIPAR2
1581  005752  062702  000200                  ADD     #200,R2
1582  005756  010237  172346                  MOV     R2,@#KIPAR3
1583  005762  012737  077406  172304          MOV     #200*256.-400+UP+RW,@#KIPDR2    ;SET KIPDR2=RW UP 200 BLOCKS
1584  005770  012737  077406  172306          MOV     #200*256.-400+UP+RW,@#KIPDR3    ;SET KIPDR3=RW UP 200 BLOCKS
1585  005776  005037  172310                  CLR     @#KIPDR4
1586  006002  012702  040000                  MOV     #40000,R2
1587  006006  012737  006024  000250          MOV     #MMABT2,@#MMVEC
1588  006014  012737  000001  177572          MOV     #1,@#SR0        ;ENABLE MEM MGMT
1589  006022  000207                  2$:     RTS     PC
1590
1591                          ;ROUTINE TO SERVICE 8 XOR 13 ABORTS
1592  006024  000240                  MMABT2: NOP
1593  006026  012702  040000                  MOV     #40000,R2
1594  006032  062737  000400  172344          ADD     #400,@#KIPAR2
1595  006040  062737  000400  172346          ADD     #400,@#KIPAR3
1596  006046  013716  177576                  MOV     @#SR2,(SP)      ;SET RETURN TO INSTRUCTION THAT ABORTED
1597  006052  012737  000001  177572          MOV     #1,@#SR0        ;ENABLE MEM MGMT
1598  006060  000002                          RTI
1599
1600                                  .SBTTL  3 XOR 9 ROUTINES
1601                          ;ROUTINE TO WRITE 3XOR9 WORST CASE NOISE TEST PATTERN
1602                          ;CALL:  MOV     BANK #,-(SP)    ;PUSH STARTING BANK # ON STACK
1603                          ;       MOV     BLKCNT,-(SP)    ;PUSH 256. WORD BLOCK COUNT ON STACK
1604                          ;       JSR     PC,.3X9         ;CALL ROUTINE
1605
1606  006062  016602  000004          .3X9:   MOV     4(SP),R2        ;GET STARTING BANK #
1607  006066  004767  177620                  JSR     PC,STMM2
1608  006072  005000                          CLR     R0
1609  006074  010003                          MOV     R0,R3
1610  006076  005103                          COM     R3              ;R0 (0) AND R3 (-1) IS THE DATA WRITTEN
1611  006100  005767  175352                  TST     PARPAT          ;BRANCH IF PARITY MEMORY PATTERN IS
1612  006104  001402                          BEQ     1$              ;NOT TO BE WRITTEN
```

```
1613
1614  006106  012700  000401              MOV     #401,R0         ;WRITE PARITY 3X9 PATTERN
1615  006112  012704  000020      1$:     MOV     #16.,R4         ;EACH LOOP WRITES 256. WORDS
1616
1617  006116  010022              2$:     MOV     R0,(R2)+
1618  006120  010022                      MOV     R0,(R2)+
1619  006122  010022                      MOV     R0,(R2)+
1620  006124  010022                      MOV     R0,(R2)+
1621
1622  006126  010022                      MOV     R0,(R2)+
1623  006130  010022                      MOV     R0,(R2)+
1624  006132  010022                      MOV     R0,(R2)+
1625  006134  010022                      MOV     R0,(R2)+
1626
1627  006136  010322                      MOV     R3,(R2)+
1628  006140  010322                      MOV     R3,(R2)+
1629  006142  010322                      MOV     R3,(R2)+
1630  006144  010322                      MOV     R3,(R2)+
1631
1632  006146  010322                      MOV     R3,(R2)+
1633  006150  010322                      MOV     R3,(R2)+
1634  006152  010322                      MOV     R3,(R2)+
1635  006154  010322                      MOV     R3,(R2)+
1636
1637  006156  005304                      DEC     R4
1638  006160  001356                      BNE     2$
1639  006162  005100                      COM     R0
1640  006164  005103                      COM     R3
1641  006166  005767  175264              TST     PARPAT          ;BRANCH IF PARITY MEMORY PATTERN IS
1642  006172  001402                      BEQ     3$              ;NOT TO BE WRITTEN
1643
1644  006174  004767  000014              JSR     PC,.XOR39       ;GO GET CONSTANTS
1645  006200  005366  000002      3$:     DEC     2(SP)           ;DECREMENT 256. WORD BLOCK COUNT
1646  006204  001342                      BNE     1$
1647  006206  012616                      MOV     (SP)+,(SP)      ;ADJUST STACK
1648  006210  012616                      MOV     (SP)+,(SP)
1649  006212  000207                      RTS     PC
1650
1651                              ;ROUTINE TO SET CONSTANTS FOR WRITING/CHECKING 3 XOR PATTERN WITH
1652                              ;PARITY. .
1653  006214  032702  000020      .XOR39: BIT     #20,R2          ;CHECK BIT 3
1654  006220  001404                      BEQ     .3IS0           ;BRANCH IF BIT 3 = 0
1655  006222  032702  002000      .3IS1:  BIT     #2000,R2        ;CHECK BIT 9
1656  006226  001404                      BEQ     .3NOT9          ;BRANCH IF BIT 9 =0
1657  006230  000407                      BR      .3IS9
1658  006232  032702  002000      .3IS0:  BIT     #2000,R2        ;CHECK BIT 9
1659  006236  001404                      BEQ     .3IS9           ;BRANCH IF 0
1660  006240  005767  172510      .3NOT9: TST     ICOUNT          ;CHECK IF NORMAL OR COMPLEMENT DATA
1661  006244  100004                      BPL     LDCOMP          ;GO LOAD COMPLEMENT CONSTANTS
1662  006246  100410                      BMI     LDNORM          ;GO LOAD NORMAL CONSTANTS
1663  006250  005767  172500      .3IS9:  TST     ICOUNT          ;CHECK IF NORMAL OR COMPLEMENT DATA
1664  006254  100005                      BPL     LDNORM          ;GO LOAD NORMAL CONSTANTS
1665  006256  012700  177777      LDCOMP: MOV     #-1,R0          ;SET COMPLEMENT CONSTANTS
1666  006262  012703  000401              MOV     #401,R3
1667  006266  000207                      RTS     PC              ;RETURN
1668  006270  012700  000401      LDNORM: MOV     #401,R0         ;LOAD NORMAL CONSTANTS
```

```
1669  006274  012703  177777              MOV     #-1,R3
1670  006300  000207                      RTS     PC
1671
1672                                ;ROUTINE TO CHECK 3 XOR 9 WORST CASE NOISE PATTERN
1673                                ;CALL:  MOV     BANK#,-(SP)     ;PUSH STARTING BANK # ONTO STACK
1674                                ;       MOV     BLKCNT,-(SP)    ;AND 256. WORD BLOACK COUNT
1675                                ;       JSR     PC,..3X9        ;CALL ROUTINE
1676
1677  006302  000240                ..3X9:  NOP
1678  006304  004767  001104              JSR     PC,CKSWR        ;GO CHECK SWITCH REGISTER
1679
1680                                ;CHECK WORST CASE PATTERN
1681  006310  016604  000002        1$:     MOV     2(SP),R4        ;GET 256. BLOCK WORD COUNT
1682  006314  016602  000004              MOV     4(SP),R2        ;GET FIRST BANK #
1683  006320  004767  177366              JSR     PC,STMM2        ;GO SET UP MEM MGMT
1684  006324  005000                      CLR     R0              ;SET CHECK WORD
1685  006326  005767  172422              TST     ICOUNT          ;IF ICOUNT IS NEG AM CHECKING COMP-
1686  006332  100001                      BPL     .+4             ;LEMENTED PATTERN
1687  006334  005100                      COM     R0              ;SO COMPLEMENT CHECK WORD
1688  006336  012705  000040        2$:     MOV     #32.,R5         ;SET 256. WORD COUNTER
1689
1690  006342  005767  175110        3$:     TST     PARPAT          ;BRANCH IF PARITY MEMORY PATTERN IS
1691  006346  001402                      BEQ     30$             ;NOT TO BE CHECKED
1692
1693  006350  004767  177640              JSR     PC,.XOR39       ;GO GET CONSTANT
1694  006354                        30$:
1695  006354  012203                      MOV     (R2)+,R3        ;GET TEST DATA
1696  006356  020003                      CMP     R0,R3           ;COMPARE WITH CHECK WORD
1697  006360  001403                      BEQ     .+10
1698  006362  005046                      CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
1699  006364  004767  172614              JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
1700                                                              ;(R0), ADDRESS=(R2)-2
1701
1702  006370  012203                      MOV     (R2)+,R3        ;GET TEST DATA
1703  006372  020003                      CMP     R0,R3           ;COMPARE WITH CHECK WORD
1704  006374  001403                      BEQ     .+10
1705  006376  005046                      CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
1706  006400  004767  172600              JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
1707                                                              ;(R0), ADDRESS=(R2)-2
1708
1709  006404  012203                      MOV     (R2)+,R3        ;GET TEST DATA
1710  006406  020003                      CMP     R0,R3           ;COMPARE WITH CHECK WORD
1711  006410  001403                      BEQ     .+10
1712  006412  005046                      CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
1713  006414  004767  172564              JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
1714                                                              ;(R0), ADDRESS=(R2)-2
1715
1716  006420  012203                      MOV     (R2)+,R3        ;GET TEST DATA
1717  006422  020003                      CMP     R0,R3           ;COMPARE WITH CHECK WORD
1718  006424  001403                      BEQ     .+10
1719  006426  005046                      CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
1720  006430  004767  172550              JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
1721                                                              ;(R0), ADDRESS=(R2)-2
1722
1723  006434  012203                      MOV     (R2)+,R3        ;GET TEST DATA
1724  006436  020003                      CMP     R0,R3           ;COMPARE WITH CHECK WORD
```

```
1725  006440  001403                        BEQ     .+10
1726  006442  005046                        CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
1727  006444  004767  172534                JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
1728                                                                  ;(R0), ADDRESS=(R2)-2
1729
1730  006450  012203                        MOV     (R2)+,R3        ;GET TEST DATA
1731  006452  020003                        CMP     R0,R3           ;COMPARE WITH CHECK WORD
1732  006454  001403                        BEQ     .+10
1733  006456  005046                        CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
1734  006460  004767  172520                JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
1735                                                                  ;(R0), ADDRESS=(R2)-2
1736
1737  006464  012203                        MOV     (R2)+,R3        ;GET TEST DATA
1738  006466  020003                        CMP     R0,R3           ;COMPARE WITH CHECK WORD
1739  006470  001403                        BEQ     .+10
1740  006472  005046                        CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
1741  006474  004767  172504                JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
1742                                                                  ;(R0), ADDRESS=(R2)-2
1743
1744  006500  012203                        MOV     (R2)+,R3        ;GET TEST DATA
1745  006502  020003                        CMP     R0,R3           ;COMPARE WITH CHECK WORD
1746  006504  001403                        BEQ     .+10
1747  006506  005046                        CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
1748  006510  004767  172470                JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
1749                                                                  ;(R0), ADDRESS=(R2)-2
1750
1751
1752  006514  005100                        COM     R0              ;COMPLEMENT CHECK WORD
1753  006516  005305                        DEC     R5              ;DECREMENT 256. WORD COUNTER
1754  006520  001310                        BNE     3$
1755  006522  005100                        COM     R0              ;COMPLEMENT CHECK WORD
1756  006524  005304                        DEC     R4              ;DECREMENT BLOCK COUNTER
1757  006526  001303                        BNE     2$
1758
1759  006530  032737  040000  177570        BIT     #40000,@#SWR    ;LOOP ON TEST?
1760  006536  001264                        BNE     1$              ;BRANCH IF LOOP ON TEST DESIRED
1761  006540  016667  000002  172220  40$:  MOV     2(SP),COUNT     ;GET # OF 256. WORD BLOCKS TO CHECK
1762  006546  016602  000004                MOV     4(SP),R2        ;GET STARTI ; BANK #
1763  006552  004767  177134                JSR     PC,STMM2        ;GO SET UP M M MGMT IF REQUIRD
1764
1765                                ;CHECK WORST CASE BIT COMPLEMENT PATTERN
1766  006556  005000                        CLR     R0
1767  006560  005767  172170                TST     ICOUNT          ;CHECK IF COMPLEMENT PATERN
1768  006564  100001                        BPL     .+4
1769  006566  005100                        COM     R0              ;COMPLEMENT CHECK WORD
1770  006570  012704  000040          4$:   MOV     #32.,R4         ;SET 256. WORD COUNTER
1771  006574  012705  000010          5$:   MOV     #8.,R5          ;SET 8 WORD COUNTER
1772  006600  005767  174652          6$:   TST     PARPAT          ;BRANCH IF PARITY MEMORY PATTERN IS
1773  006604  001402                        BEQ     60$             ;NOT TO BE CHECKED
1774  006606  004767  177402                JSR     PC,.XOR39
1775  006612  012203                  60$:  MOV     (R2)+,R3        ;GET DATA
1776  006614  020003                        CMP     R0,R3           ;CHECK DATA
1777  006616  001403                        BEQ     .+10
1778  006620  005046                        CLR     -(SP)
1779  006622  004767  172356                JSR     PC,ERROR
1780  006626  005100                        COM     R0              ;COMPLEMENT CHECK WORD
```

```
1781  006630  005142                          COM     -(R2)           ;COMPLEMENT TEST DATA
1782  006632  012203                          MOV     (R2)+,R3        ;GET DATA
1783  006634  020003                          CMP     R0,R3           ;CHECK
1784  006636  001403                          BEQ     .+10
1785  006640  005046                          CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
1786  006642  004767  172336                  JSR     PC,ERROR
1787  006646  005100                          COM     R0              ;COMPLEMENT CHECK WORD
1788  006650  005162  177776                  COM     -2(R2)          ;RESTORE DATA
1789  006654  005305                          DEC     R5              ;DECREMENT 4 WORD COUNTER
1790  006656  001350                          BNE     6$
1791  006660  005100                          COM     R0              ;COMPLEMENT CHECK WORD
1792  006662  005304                          DEC     R4              ;DECREMENT 256. WORD COUNTER
1793  006664  001343                          BNE     5$
1794  006666  005100                          COM     R0              ;COMPLEMENT CHECK WORD
1795  006670  005367  172072                  DEC     COUNT           ;DECREMENT BLOCK COUNTER
1796  006674  001335                          BNE     4$
1797
1798  006676  016602  000004                  MOV     4(SP),R2        ;GET BANK #
1799  006702  004767  177004                  JSR     PC,STMM2
1800  006706  016603  000002                  MOV     2(SP),R3        ;GET BLOCK COUNT
1801  006712  032737  040000  177570          BIT     #40000,@#SWR    ;LOOP ON TEST
1802  006720  001307                          BNE     40$             ;BRANCH IF LOOP ON TEST
1803  006722  006367  172026                  ASL     ICOUNT
1804  006726  102402                          BVS     7$
1805  006730  000167  177354                  JMP     1$
1806  006734  012705  000020          7$:     MOV     #16.,R5         ;COMPLEMENT PATTERN
1807  006740  011200                  10$:    MOV     (R2),R0         ;GET 1ST DATA WORD
1808  006742  016204  000020                  MOV     20(R2),R4       ;GET 9TH DATA WORD
1809  006746  110422                          MOVB    R4,(R2)+        ;SWAP WORDS 1-8
1810  006750  110422                          MOVB    R4,(R2)+        ;WITH 9-16
1811  006752  110422                          MOVB    R4,(R2)+
1812  006754  110422                          MOVB    R4,(R2)+
1813  006756  110422                          MOVB    R4,(R2)+
1814  006760  110422                          MOVB    R4,(R2)+
1815  006762  110422                          MOVB    R4,(R2)+
1816  006764  110422                          MOVB    R4,(R2)+
1817  006766  110422                          MOVB    R4,(R2)+
1818  006770  110422                          MOVB    R4,(R2)+
1819  006772  110422                          MOVB    R4,(R2)+
1820  006774  110422                          MOVB    R4,(R2)+
1821  006776  110422                          MOVB    R4,(R2)+
1822  007000  110422                          MOVB    R4,(R2)+
1823  007002  110422                          MOVB    R4,(R2)+
1824  007004  110422                          MOVB    R4,(R2)+
1825  007006  110022                          MOVB    R0,(R2)+        ;AND VICE VERSA
1826  007010  110022                          MOVB    R0,(R2)+
1827  007012  110022                          MOVB    R0,(R2)+
1828  007014  110022                          MOVB    R0,(R2)+
1829  007016  110022                          MOVB    R0,(R2)+
1830  007020  110022                          MOVB    R0,(R2)+
1831  007022  110022                          MOVB    R0,(R2)+
1832  007024  110022                          MOVB    R0,(R2)+
1833  007026  110022                          MOVB    R0,(R2)+
1834  007030  110022                          MOVB    R0,(R2)+
1835  007032  110022                          MOVB    R0,(R2)+
1836  007034  110022                          MOVB    R0,(R2)+
```

```
 1837  007036  110022                        MOVB    R0,(R2)+
 1838  007040  110022                        MOVB    R0,(R2)+
 1839  007042  110022                        MOVB    R0,(R2)+
 1840  007044  110022                        MOVB    R0,(R2)+
 1841  007046  005305                        DEC     R5
 1842  007050  001333                        BNE     10$
 1843  007052  005303                        DEC     R3
 1844  007054  001327                        BNE     7$
 1845
 1846  007056  005767  171672                TST     ICOUNT
 1847  007062  001402                        BEQ     11$
 1848  007064  000167  177220                JMP     1$
 1849  007070  012616              11$:      MOV     (SP)+,(SP)
 1850  007072  012616                        MOV     (SP)+,(SP)
 1851  007074  000207                        RTS     PC
 1852
 1853                                  ;ROUTINE TO WRITE 8 XOR 13 WORST CASE NOISE TEST PATTERN
 1854                                          .SBTTL  8 XOR 13 ROUTINES
 1855                                  ;CALL:  MOV     BANK #,-(SP)
 1856                                  ;       MOV     #4*BANKS,-(SP)
 1857                                  ;       JSR     PC,..8X13
 1858
 1859  007076  012616              .8X13:    MOV     (SP)+,(SP)      ;ADJUST STACK
 1860  007100  012616                        MOV     (SP)+,(SP)
 1861  007102  000207                        RTS     PC
 1862
 1863                                  ;ROUTINE TO CHECK 8 XOR 13 WORST CASE NOISE TEST PATTERN
 1864                                  ;CALL:
 1865                                  ;       MOV     BANK#,-(SP)     ;PUSH FIRST BANK # ON THE STACK
 1866                                  ;       MOV     #BANKS,-(SP)    ;PUSH # OF 4K BANKS TO CHECK ON THE STACK
 1867                                  ;       JSR     PC,..8X13       ;CALL ROUTINE
 1868
 1869  007104  012616              ..8X13:   MOV     (SP)+,(SP)
 1870  007106  012616                        MOV     (SP)+,(SP)
 1871  007110  000207                        RTS     PC              ;RETURN
 1872
 1873                                          .SBTTL  ROTATING 1'S & 0'S ROUTINES
 1874                                  ;ROUTINE TO CHECK ROTATING '0' BIT THROUGH FILD OF 1'S
 1875                                  ;CALL:  MOV     BANK#,-(SP)     ;SET STARTING BANK #
 1876                                  ;       MOV     BLKCNT,-(SP)    ;SET 256. WORD BLOCK COUNT
 1877                                  ;       JSR     PC,..ROTO       ;CALL ROUITNE
 1878
 1879  007112  004767  000276          .ROTO:  JSR     PC,CK3WR        ;GO CHECK SWITCHES
 1880  007116  016604  000002                  MOV     2(SP),R4        ;GET 256. WORD BLOCK COUNT
 1881  007122  016602  000004                  MOV     4(SP),R2        ;GET FIRST BANK #
 1882  007126  004767  176560                  JSR     PC,STMM2        ;GO SET UP MEM MGMT (IF AVAIL)
 1883  007132  012700  177777                  MOV     #-1,R0          ;SET CHECK WORD
 1884
 1885  007136  012705  000400          1$:     MOV     #256.,R5        ;SET 256. WORD COUNT
 1886  007142  000241                  2$:     CLC                     ;CLEAR CARRY BIT IN PSW
 1887  007144  004767  000124                  JSR     PC,ROTATE
 1888  007150  016203  177776                  MOV     -2(R2),R3       ;GET RESULT
 1889  007154  103402                        BCS     3$              ;BRANCH IF 'C' BIT WAS SET
 1890  007156  020003                        CMP     R0,R3           ;CHECK RESULT
 1891  007160  001403                        BEQ     4$
 1892  007162  005046                  3$:     CLR     -(SP)           ;ERROR! COULD NOT ROTATE '0' BIT
```

```
1893  007164  004767  172014                  JSR     PC,ERROR        ;THROUGH ADDRESS IN R2
1894  007170  005305                  4$:     DEC     R5              ;DECREMENT 256. WORD COUNT
1895  007172  001363                          BNE     2$              ;LOOP UNTIL DONE
1896  007174  005304                          DEC     R4              ;DECREMENT 256. WORD BLOCK COUNT
1897  007176  001357                          BNE     1$              ;LOOP UNTIL DONE
1898  007200  012616                          MOV     (SP)+,(SP)      ;POP CONSTANTS OFF THE STACK
1899  007202  012616                          MOV     (SP)+,(SP)
1900  007204  000207                          RTS     PC              ;RETURN TO CALLER
1901
1902                                  ;ROUTINE TO CHECK ROTATING '1' BIT THROUGH A FIELD OF 0'S
1903                                  ;CALL:  MOV     BANK#,-(SP)     ;SET STARTING BANK #
1904                                  ;       MOV     BLKCNT,-(SP)    ;SET # OF 256. WORD BLOCKS TO CHECK
1905                                  ;       JSR     PC,.ROT1        ;CALL ROUTINE
1906
1907  007206  004767  000202          .ROT1:  JSR     PC,CKSWR        ;GO CHECK SWITCHES
1908  007212  016604  000002                  MOV     2(SP),R4        ;GET # OF 256. WORD BLOCKS TO CHECK
1909  007216  016602  000004                  MOV     4(SP),R2        ;GET STARTING BANK #
1910  007222  004767  176464                  JSR     PC,STMM2        ;GO SET UP MEM MGMT (IF AVAIL)
1911  007226  005000                          CLR     R0              ;SET CHECK WORD
1912
1913  007230  012705  000400          1$:     MOV     #256.,R5        ;SET 256. WORD COUNTER
1914  007234  000261                  2$:     SEC                     ;SET 'C' BIT IN PSW
1915  007236  004767  000032                  JSR     PC,ROTATE       ;GO ROTATE '1' BIT
1916  007242  016203  177776                  MOV     -2(R2),R3       ;GET RESULT
1917  007246  103002                          BCC     3$              ;BRANCH IF 'C' IS CLEAR
1918  007250  020003                          CMP     R0,R3           ;CHECK RESULT
1919  007252  001401                          BEQ     .+4
1920  007254  104400                  3$:     HLT                     ;ERROR! COULD NOT ROTATE '1' BIT
1921                                                                  ;THROUGH ADDRESS IN R2
1922  007256  005305                          DEC     R5              ;DECREMENT 256. WORD COUNT
1923  007260  001365                          BNE     2$
1924  007262  005304                          DEC     R4              ;DECREMENT 256. WORD BLOCK COUNT
1925  007264  001361                          BNE     1$
1926  007266  012616                          MOV     (SP)+,(SP)      ;ADJUST RETURN ADDRESS
1927  007270  012616                          MOV     (SP)+,(SP)
1928  007272  000207                          RTS     PC              ;RETURN TO CALLER
1929
1930                                  ;ROUTINE TO ROTATE 'C' BIT THROUGH A MEMORY LOCATION.
1931  007274  106112                  ROTATE: ROLB    (R2)            ;(R2)=177776 OR 000001
1932  007276  106112                          ROLB    (R2)            ;(R2)=177775 OR 000002
1933  007300  106112                          ROLB    (R2)            ;(R2)=177773 OR 000004
1934  007302  106112                          ROLB    (R2)            ;(R2)=177767 OR 000010
1935  007304  106112                          ROLB    (R2)            ;(R2)=177757 OR 000020
1936  007306  106112                          ROLB    (R2)            ;(R2)=177737 OR 000040
1937  007310  106112                          ROLB    (R2)            ;(R2)=177677 OR 000100
1938  007312  106112                          ROLB    (R2)            ;(R2)=177777 OR 000000
1939  007314  106122                          ROLB    (R2)+           ;(R2)=177577 OR 000200
1940  007316  106112                          ROLB    (R2)            ;(R2)=177377 OR 000400
1941  007320  106112                          ROLB    (R2)            ;(R2)=176777 OR 001000
1942  007322  106112                          ROLB    (R2)            ;(R2)=175777 OR 002000
1943  007324  106112                          ROLB    (R2)            ;(R2)=173777 OR 004000
1944  007326  106112                          ROLB    (R2)            ;(R2)=167777 OR 010000
1945  007330  106112                          ROLB    (R2)            ;(R2)=157777 OR 020000
1946  007332  106112                          ROLB    (R2)            ;(R2)=137777 OR 040000
1947  007334  106112                          ROLB    (R2)            ;(R2)=077777 OR 100000
1948  007336  106122                          ROLB    (R2)+           ;(R2)=177777 OR 000000
```

```
1949  007340  000207                          RTS     PC              ;RETURN
1950
1951                                  ;ROUTINE TO WRITE ONE WORD PATTERN INTO MEMORY
1952                                  ;CALL:  MOV     BANK#,-(SP)     ;PUSH STARTING BANK # ONTO STACK
1953                                  ;       MOV     BLKCNT,-(SP)    ;AND 128. WORD BLOCK COUNT
1954                                  ;       JSR     PC,WRTPAT       ;CALL ROUTINE
1955
1956  007342  016604  000002          WRTPAT: MOV     2(SP),R4        ;GET BLOCK COUNT
1957  007346  016602  000004                  MOV     4(SP),R2        ;GET STARTING BANK #
1958  007352  004767  176334                  JSR     PC,STMM2        ;GO SET UP MEM MGMT
1959  007356  012700                          MOV     (PC)+,R0        ;GET USER CONSTANT
1960  007360  000000                  .CONST: 0
1961  007362  012703  000100          1$:     MOV     #64.,R3         ;SET 256. WORD COUNTER
1962  007366  010022                  2$:     MOV     R0,(R2)+        ;WRITE 256. WORDS
1963  007370  010022                          MOV     R0,(R2)+
1964  007372  010022                          MOV     R0,(R2)+
1965  007374  010022                          MOV     R0,(R2)+
1966  007376  005303                          DEC     R3              ;DECREMENT 256. WORD COUNTER
1967  007400  001372                          BNE     2$              ;LOOP UNTIL 256. WORDS HAVE BEEN WRITTEN
1968  007402  005304                          DEC     R4              ;DECREMENT BLOCK COUNT
1969  007404  001366                          BNE     1$
1970  007406  012616                          MOV     (SP)+,(SP)      ;ADJUST STACK
1971  007410  012616                          MOV     (SP)+,(SP)
1972  007412  000207                          RTS     PC
1973
1974
1975                                  ;ROUTINE TO CHECK THE SWITCH REGISTER
1976                                  ;CHECK SWITCH 9: IF SET, LOAD ERROR COUNT INTO THE DISPLAY REGISTER;
1977                                  ;IF NOT SET, LOAD PASS COUNT INTO THE DISPLAY REGISTER
1978  007414  042767  017777  171336  CKSWR:  BIC     #17777,LDDISP   ;SAVE RELOCATION BITS
1979  007422  032737  000400  177570          BIT     #BIT8,@#SWR     ;CHECK SWITCH 8
1980  007430  001402                          BEQ     10$             ;BRANCH IF SET
1981  007432  004767  000464                  JSR     PC,REL24K       ;GO RELOCATE PROGRAM BACK TO 4K AND STOP
1982  007436  032737  001000  177570  10$:    BIT     #BIT9,@#SWR     ;SWITCH 9 SET ?
1983  007444  001404                          BEQ     1$
1984  007446  056767  171304  171304          BIS     ERCNT,LDDISP    ;LOAD ERROR COUNT
1985  007454  000403                          BR      2$
1986  007456  056767  171270  171274  1$:     BIS     ICNT,LDDISP     ;LOAD PASS COUNT
1987  007464  016737  171270  177570  2$:     MOV     LDDISP,@#DISPLAY;LOAD THE DISPLAY REGISTER
1988  007472  012767  040177  171254          MOV     #040177,ICOUNT  ;LOAD ITERATION COUNT WORD
1989  007500  032737  004000  177570          BIT     #4000,@#SWR     ;CHECK SW11
1990  007506  001402                          BEQ     .+6
1991  007510  105067  171240                  CLRB    ICOUNT          ;ICOUNT =040000 IF SW11 =1
1992  007514  000207                          RTS     PC
1993
1994                                  ;MESSAGES
1995  007516  005015  047524  051040  RESLDR: .ASCIZ  <15><12>'TO RESTORE LOADERS START AT 162'<15><12>
1996  007524  051505  047524  042522
1997  007532  046040  040517  042504
1998  007540  051522  051440  040524
1999  007546  052122  040440  020124
2000  007554  033061  006462  000012
2001  007562  005015  047105  041101  PARITY: .ASCIZ  <15><12> 'ENABLE PARITY? 1/0=YES/NO '
2002  007570  042514  050040  051101
2003  007576  052111  037531  030440
2004  007604  030057  054475  051505
```

```
 2005  007612  047057  020117     000
 2006  007617     015  051412  040524  STBANK: .ASCIZ  <15><12>'STARTING BANK #(8)? '
 2007  007624  052122  047111  020107
 2008  007632  040502  045516  021440
 2009  007640  034050  037451  000040
 2010  007646  005015  020043  043117  BANKS:  .ASCIZ  <15><12>'# OF 4K BANKS TO TEST(8)? '
 2011  007654  032040  020113  040502
 2012  007662  045516  020123  047524
 2013  007670  052040  051505  024124
 2014  007676  024470  020077     000
 2015  007703     015  050012  052101  PAT:    .ASCIZ  <15><12>'PATTERN #? '
 2016  007710  042524  047122  021440
 2017  007716  020077     000
 2018  007721     015  037412     000  QUEST:  .ASCIZ  <15><12>'?'
 2019  007725     015  052012  050131  CONST:  .ASCIZ  <15><12>'TYPE CONSTANT'
 2020  007732  020105  047503  051516
 2021  007740  040524  052116     000
 2022  007745     015  044412  050116  PRG3M:  .ASCIZ  <15><12>'INPUT # OF 256. WORD BLOCKS TO TEST INSTEAD OF'
 2023  007752  052125  021440  047440
 2024  007760  020106  032462  027066
 2025  007766  053440  051117  020104
 2026  007774  046102  041517  051513
 2027  010002  052040  020117  042524
 2028  010010  052123  044440  051516
 2029  010016  042524  042101  047440
 2030  010024  000106
 2031  010026  005015  054524  042520  PRG4M:  .ASCIZ  <15><12>'TYPE ADDRESS'
 2032  010034  040440  042104  042522
 2033  010042  051523     000
 2034  010045     015  052012  020117  RELOCM: .ASCIZ  <15><12>'TO RESTORE PROGRAM START AT '
 2035  010052  042522  052123  051117
 2036  010060  020105  051120  043517
 2037  010066  040522  020115  052123
 2038  010074  051101  020124  052101
 2039  010102  000040
 2040  010104  000052                  ASTERISK:.ASCIZ '*'
 2041  010106  042504  045115  020101  ENDMSG: .ASCIZ  'DEMJA DONE!'
 2042  010114  047504  042516  000041
 2043                                          .EVEN
 2044
 2045                                  ;ROUTINE TO RELOCATE PROGRAM BACK TO 0
 2046  010122  010700                  REL24K: MOV     PC,R0                   ;FORM BASE ADDRESS WHERE CODE
 2047  010124  042700  017777                  BIC     #17777,R0               ;IS RELOCATED
 2048  010130  010067  000004                  MOV     R0,1$                   ;PUT FROM ADDRESS INTO SUBROUTINE CALL
 2049  010134  004567  174112                  JSR     R5,RELOC                ;RELOCATE CODE TO
 2050  010140  000000                  1$:     0                               ;LOWEST 4K
 2051  010142  000000                          0
 2052  010144  012706  000500                  MOV     #STKPTR,SP              ;SET STACK PTR
 2053  010150  042737  100000  000760          BIC     #100000,@#LDDISP        ;CLEAR RELOCATION INDICATOR
 2054  010156  013737  000760  177570          MOV     @#LDDISP,@#DISPLAY      ;LOAD DISPLAY REGISTER
 2055  010164  005037  000764                  CLR     @#RELOCF                ;CLEAR RELOCATION FACTOR
 2056  010170  000005                          RESET                           ;DISABLE MEM MGMT
 2057  010172  000137  000162                  JMP     @#PONE                  ;RESTORE LOADERS & HALT
 2058
 2059  010176                          LOCAR:
 2060          000001                          .END
```

```
ADRESS  002327            686     871*   1436
ASTERI  010104           1248    2040*
BANKS   007646           2010*
BEGIN1  003440           1103    1105    1111*   1249
BELL    001527            705     721*
BIT12 = 010000            401*    915
BIT13 = 020000            400*   1410
BIT14 = 040000            399*
BIT15 = 000100            398*
BIT6  = 000100            404*
BIT8  = 000400            403*   1979
BIT9  = 001000            402*    665    1982
BPTVEC= 000014            371*
C     = 000001            351*
CKSWR   007414           1113    1678    1879    1907    1978*
CNTRL = 177746           1511*
CONST   007725           2019*
COUNT   000766            586*   1761*   1795*
CRLF    000100            469*    700     943
DIGITS  002351            652     682     731     875*   1424
DIGTAB  002312            769     864*   1432
DISPAR  005430           1484    1502*
DISPLA= 177570            392*    669*    903*   1987*   2054*
DONE    004210           1246    1250*
DONE0   002620            929     939*
DONE1   002724            948     972*
DONE10  003700           1170    1175*
DONE3   003026            997    1002*
DONE4   003114           1008    1019*
DONE4A  003164           1022    1035*
DONE4B  003234           1038    1052*
DONE4C  003302           1063    1065    1070*
DONE4D  003414           1070    1101*
DONE6   003474           1117    1126*   1153
DONE7   003522           1128    1134*
DONE8   003614           1149    1151    1156*
DONE9   003642           1157    1164*
EMTVEC= 000030            374*
END     004152           1239*
ENDCT   004170           1106*   1244*
ENDMSG  010106           1252    2041*
ERCNT   000756            579*    662     664*    668     901*   1984
ERROR   001204            461     661*   1699    1706    1713    1720    1727    1734    1741    1748    1779    1786    1893
ERRPC   001475            680     716*   1417
ERRTRP  001126            458     640*   1134    1166    1215    1309
ERRVEC= 000004            367*    821*    917*    929*    948*   1008*   1022*   1038*   1070*   1128*   1134*   1157*   1166*
                         1181*   1215*   1271*   1304*   1309*   1344*   1485    1486*   1501*
ESTRB   005235           1456    1465*
FINISH  004246           1256    1262*
FPEVEC= 000244            379*
HCRNT   005302           1459    1472*
HLT   = 104400            432*    956     963    1017    1048    1081    1088    1096    1202    1284    1355    1372    1920
ICNT    000752            577*    908*   1242*   1245    1444    1514    1986
ICOUNT  000754            578*   1660    1663    1685    1767   1903*   1846    1988*   1991*
ICTVEC= 000020            372*
KIPARO= 172340            421*    529     778    1539*
```

```
kIPAR1= 172342           422*   779    980   1006*  1209*  1540*  1548*  1556*
kIPAR2= 172344           423*   780   1187*  1210*  1537*  1580*  1594*
kIPAR3= 172346           424*   781   1582*  1595*
kIPAR4= 172350           425*   782
kIPAR5= 172352           426*   783
kIPAR6= 172354           427*   784
kIPAR7= 172356           428*   542    785   1541*
kIPDR0= 172300           412*   524   1185*  1349*  1534*
kIPDR1= 172302           413*  1535*
kIPDR2= 172304           414*  1186*  1536*  1583*
kIPDR3= 172306           415*  1584*
kIPDR4= 172310           416*  1585*
kIPDR5= 172312           417*
kIPDR6= 172314           418*
kIPDR7= 172316           419*   547   1538*
KM    = 000000           358*
LCRNT   005266          1458   1470*
LDCOMP  006256          1661   1665*
LDDISP  000760           580*   667*   668*   669    902*   903   1978*  1984*  1986*  1997   2053*  2054
LDMMO   005512           931    951   1024   1040   1072   1184   1348   1530*  1577
LDNORM  006270          1662   1664   1668*
LDDAR   010176           830    844   2059*
LDDFLG  002110           815*   817    835*   839    853*
LOGICA  004236           464   1104   1258*
LST     002322           870*   942
LSTLOC  002244           834*   850*
LSTRB   005252          1457   1468*
MAINTR= 177750          1512*  1516*
MARGIN  005464          1118   1514*
MARMSG  005317          1451   1475*
MARTBL  005210          1446   1449   1454*
MEMERR= 177744          1395*  1406   1409*
MEMHI = 177742          1394*  1408
MEMLO = 177740          1393*  1407
MMABT0  005614           932   1007   1025   1073   1350   1547*
MMABT1  005646           990   1057   1555*
MMABT2  006024          1587   1592*
MMAVA   000762           519    540    581*   736    914*   919*   923*   978   1004   1116   1146   1192   1207
                        1346   1502   1531   1569
MMVEC = 000250           380*   932*   990*  1007*  1025*  1057*  1073*  1350*  1587*
MRGNTB  005504          1445   1516   1520*
N     = 000010           354*
NOFIND  004723          1371   1386*
NORMAL  005226          1454   1460   1463*
O2A     002014           694    699    790*  1319   1420   1441
PARAVA  005402           553   1148   1492*  1498*  1504*
PARCSR= 172100          1480*  1489
PARERR  004702          1339   1383*  1414
PARITY  007562          2001*
PARPAT  003456          1115*  1150   1152*  1611   1641   1690   1772
PARREG  001531           722*  1439
PARTAB  001774           746    778*
PARVEC= 000114          1343*  1481*  1487*  1488*  1505*  1506*
PASSMG  002363           878*
PASSNM  002370           879*
PAT     007703          2015*
```

```
PC     =.000007          342#  471*  476*  496*  508*  518*  532*  555*  557*  560*  595*  622*  630*
                         636*  643   648*  676*  678*  684*  694*  699*  700*  701*  730*  773*  776*
                         791*  810*  812*  819*  936*  841*  849   854*  898*  906*  924*  930   931*
                         940*  943*  946   951*  973   1003  1023  1024* 1039  1040* 1056  1071  1072*
                        1112* 1113* 1114* 1118* 1127  1131* 1137* 1140  1143* 1158  1161* 1165  1172*
                        1180  1184* 1221* 1222  1225* 1232* 1233  1236* 1240  1243  1257* 1258* 1319*
                        1322  1325* 1340* 1348* 1361* 1420* 1441* 1442* 1491* 1507* 1518* 1544* 1575*
                        1577* 1589* 1607* 1644* 1649* 1667* 1670* 1678* 1683* 1693* 1699* 1706* 1713*
                        1720* 1727* 1734* 1741* 1748* 1763* 1774* 1779* 1786* 1799* 1851* 1861* 1871*
                        1879* 1882* 1887* 1893* 1900* 1907* 1910* 1915* 1928* 1949* 1958* 1959  1972*
                        1981* 1992* 2046
PDWN     000502          517#   559    910
PEFLG    004524          689    905*  1341# 1353* 1358* 1365  1398  1402*
PENFLG   004525          687   1342# 1369*
PERSTR   002454          907#  1362   1401  1452
PFSTK    000564          533#   539
PFVEC =  000024          373#   534*   559*  910*  911*
PIRQ  =  177772          385#
PIRVEC=  000240          378#
PKM   =  000000          361#
PLACE    002374          881#  1196
PONE     000162          475#  2057
PRG3M    007745         2022#
PRG4M    010026         2031#
PRTY4 =  000200          357#
PRTY7 =  000340          356#   462
PSM   =  010000          362#
PSW   =  177776          383#   594
PTWO     000200          478    480#
PUM   =  030000          363#
PUP      000576          534    538#
PWRFAI   000730          565    569#
QUEST    007721         2018#
RANTST   003702         1180#
RECDAT   001514          697    719#
REG   =  004000          364#
RELFL    000110          472#
RELOC    004252         1267#  1312   2049
RELOCF   000764          585#   593    728   729   774   775   909*  1294  1419  2055*
RELOCM   010045         1316   2034#
RELOCP   004354         1303#
REL24K   010122         1318   1981   2046#
RESLDR   007516          900   1995#
RESVEC=  000010          368#
ROTATE   007274         1887   1915   1931#
ROTO     004054         1218#
ROTI     004114         1229#
PW    =  000006          436#  1186   1534  1535  1538  1583  1584
RO    =%.000000          335#   495    502*  524*  527   529*  530   542*  545*  547*  548*  612   613*
                         616    619*   693   709   747*  750*  755*  759*  761*  771*  792*  795*  902*
                         804*   806*   820*  822   825   827*  828   831   949*  950*  952*  954   961
                        1010*  1011*  1012* 1013* 1015  1027* 1028* 1030  1042* 1043* 1046  1059* 1060*
                        1062   1079   1086  1091* 1094  1135* 1136* 1137  1142  1197* 1199  1255* 1258
                        1267*  1274   1280  1303* 1305* 1307  1310* 1311  1331  1332* 1333* 1334* 1335*
                        1336*  1337*  1351* 1354  1424* 1432* 1444* 1445* 1446* 1447  1489* 1496* 1514*
                        1516   1608*  1609  1614* 1617  1618  1619  1620  1622  1623  1624  1625  1639*
```

```
                        1665*  1668*  1684*  1687*  1696   1703   1710   1717   1724   1731   1738   1745   1752*
                        1755*  1766*  1769*  1776   1780*  1783   1787*  1791*  1794*  1807*  1825   1826   1827
                        1828   1829   1830   1831   1932   1833   1834   1835   1836   1837   1838   1839   1840
                        1883*  1890   1911*  1918   1959*  1962   1963   1964   1965   2046*  2047*  2048
R1      =%000001         336*   494    503*   735*   740*   741*   742*   743*   744*   745*   754*   758*   763*
                         930*   946*   973*  1003*  1023*  1039*  1056*  1071*  1127*  1140*  1158*  1165*  1190*
                        1222*  1233*  1240*  1333   1407*  1428*
R10     =%000000         343#
R11     =%000001         344#
R12     =%000002         345#
R13     =%000003         346#
R14     =%000004         347#
R15     =%000005         348#
R2      =%000002         337#   493    504*   525*   526    528*   543*   544    549*   647*   677*   683*   709*
                         710*   733*   734    746*   748*   752*   757*   764*   794*   796*   797*   798    801*
                         803*   805*   807*   829*   832*   845*   847*   933*   934    935*   947*   949    953
                         958*   959*   960    987    992*   993    996*   998    1009*  1010   1014   1026*  1030*
                        1041*  1045   1058*  1062*  1064   1074*  1078   1083*  1084*  1085   1093   1188*  1191*
                        1192*  1198   1201   1203   1212*  1268*  1269   1274*  1275   1279   1280   1294*  1295*
                        1296*  1334   1345*  1351   1367*  1368*  1408*  1429*  1490*  1498   1499*  1515*  1517*
                        1547*  1555*  1567   1571*  1572*  1573*  1574*  1578*  1579*  1580   1581*  1582   1586*
                        1593*  1606*  1617*  1618*  1619*  1620*  1622*  1623*  1624*  1625*  1627*  1628*  1629*
                        1630*  1632*  1633*  1634*  1635*  1653   1655   1658   1682*  1695   1702   1709   1716
                        1723   1730   1737   1744   1762*  1775   1781*  1782   1788*  1798*  1807   1808   1809*
                        1810*  1811*  1812*  1813*  1814*  1815*  1816*  1817*  1818*  1819*  1820*  1821*  1822*
                        1823*  1824*  1825*  1826*  1827*  1828*  1829*  1830*  1831*  1832*  1833*  1834*  1835*
                        1836*  1837*  1838*  1839*  1840*  1881*  1888   1909*  1916   1931*  1932*  1933*  1934*
                        1935*  1936*  1937*  1938*  1939*  1940*  1941*  1942*  1943*  1944*  1945*  1946*  1947*
                        1948*  1957*  1962*  1963*  1964*  1965*
R3      =%000003         338#   492    505*   526*   531*   544*   546*   698    732*   749*   753*   756*   765*
                         770*   793*   808*   830*   831*   834    934*   935    936*   953*   954    960*   961
                         980*   981*   982*   983*   984*   985*   986*   989*   993*   994*   995*   996    1014*
                        1015   1045*  1046   1078*  1079   1085*  1086   1093*  1094   1198*  1199   1269*  1270*
                        1273   1275   1277*  1278   1282   1335   1354*  1427*  1430*  1432   1609*  1610*  1627
                        1628   1629   1630   1632   1633   1634   1635   1640*  1666*  1669*  1695*  1696   1702*
                        1703   1709*  1710   1716*  1717   1723*  1724   1730*  1731   1737*  1738   1744*  1745
                        1775*  1776   1782*  1783   1800*  1843*  1888*  1890   1916*  1918   1961*  1966*
R4      =%000004         339#   491    506*   731*   844*   846    1029*  1031*  1044*  1049*  1061*  1066*  1075
                        1089*  1092*  1097*  1168*  1169*  1171   1189*  1191   1192   1197   1272*  1292*  1336
                        1425*  1433*  1615*  1637*  1681*  1756*  1770*  1792*  1808*  1909   1810   1811   1912
                        1813   1814   1815   1816   1817   1818   1819   1820   1821   1822   1823   1824   1880*
                        1856*  1908*  1924*  1956*  1968*
R5      =%000005         340#   469*   490    507*   564*   592    597*   649*   651*   679*   681*   685*   691*
                         696*   704*   734*   735    751*   752    766*   768*   799*   842*   846*   949*   851*
                         899*   941*  1190*  1193*  1196*  1204*  1247*  1251*  1267   1268   1277   1295   1296
                        1312*  1315*  1337   1338*  1370*  1413*  1416*  1421*  1426*  1431*  1435*  1438*  1440*
                        1448*  1450*  1688*  1753*  1771*  1789*  1806*  1841*  1885*  1894*  1913*  1922*  2049*
SAVPC2  000112           473#
SAVR0   004666          1331*  1332   1376#
SAVR1   004670          1377#
SAVR2   004672          1378#
SAVR3   004674          1379#
SAVR4   004676          1380#
SAVR5   004700          1381#
SCOPE = 104000           433#
SLR   = 177774           384#
```

```
SM     = 040000          359#
SP     =%000006          341#    475*    480*    489     490*    491*    492*    493*    494*    495*    501     502     503
                         504     505     506     507     517*    521*    522*    523*    527*    530*    532     539*    545
                         548     550     551     552     558     594*    612*    613     614*    616*    618     619     623
                         625*    628*    635     646     647     677     683     693*    698*    792     811*    824     897*
                         904*    939*    972*    987*    988*    989     1035    1052    1101    1111*   1126*   1129*   1130*
                        1135    1139*   1141*   1142*   1152    1156*   1159*   1160*   1164*   1167*   1171*   1214*   1219*
                        1220*   1223*   1224*   1230*   1231*   1234*   1235*   1241*   1291    1317*   1318*   1320    1321*
                        1322*   1324*   1353    1359*   1367    1369    1402    1406*   1409    1418*   1419*   1440    1485*
                        1501    1549*   1558*   1563*   1596*   1606    1645*   1647*   1648*   1681    1682    1698*   1705*
                        1712*   1719*   1726*   1733*   1740*   1747*   1761    1762    1778*   1785*   1798    1800    1949*
                        1850*   1859*   1860*   1869*   1870*   1880    1881    1892*   1898*   1899*   1908    1909    1926*
                        1927*   1956    1957    1970*   1971*   2052*
SPACE1  002361           877#    1422    1442
SR0    = 177572          407#     521     552*    738     918*    1542*   1550*   1551*   1559*   1562*   1588*   1597*
SR1    = 177574          408#
SR2    = 177576          409#     522     551*   1549    1558    1596
SR3    = 172516          410#     523     550*    920*    921    1533*
START   002376           481     896#
START1  002440           896      904#    906     1262
STBANK  007617          2006#
STKPTR= 000500            395#    897     904     939     972     1111    1126    1139    1156    1164    1214    1241    1359
                         2052
STMM2   005712          1567#   1607    1683    1763    1799    1882    1910    1958
SWR    = 177570           391#    640     665     670     674     702     706     915    1403    1410    1483    1759    1801
                         1979    1982    1989
T      = 000020          355#
TBITVE= 000014           369#
TKB    = 177562          388#
TKS    = 177560          387#     517     558*
TKVEC  = 000060          376#
TPB    = 177566          390#
TPS    = 177564          389#    1253
TPVEC  = 000064          377#
TRAPVE= 000034           375#     460
TRAP4   001460           650      713#
TRTVEC= 000014           370#
TYPIT   001110           622      630     633#
UBREAK= 177770           386#
UM     = 140000          360#
UP     = 000000          437#    1186    1534    1535    1538    1583    1584
V      = 000002          352#    1563
WDS.25  003540          1138#    1220    1224    1231    1235
WRTPAT  007342          1221     1232    1956#
WRTUP   002560           929#
XMTDAT  001501           692      717#
Z      = 000004          353#
$CRLF   000746           470      572#
$FILL   001021           606#
$FORMO  001554           648      678     684     728#    940
$LDR    002112           817#    898
$LDRM   002264           852     858#
$LDR1   002262           828*    842     856#
$NULL   001020           605#    625
$PRINT  000770           469      564     591#    649     651     679     681     685     691     696     704     799     951
                         899      941    1247    1251    1315    1338    1370    1413    1416    1421    1435    1438    1448
```

```
                        1450
$RESTR  000300           501*    557     701     773     810
$RLDR   002210           476     839*   1257
$SAVR   000254           489*    518     676     730     791
$TKFLG  001023           609*
$TPB    001026           611*    635*
$TPFLG  001022           607*
$TPS    001024           610*    633
$$$TAG= 016066          2060*
.       = 010176           439*    454*    460*    463*    466*    468*    474*    479*    482*    512*    576*    634     641
                           671     707     726*    750     823     862*    880*   1047    1095    1254    1290   1390*   1477*
                          1686    1697    1704    1711    1718    1725    1732    1739    1746    1768    1777    1784    1919
                          1990    2060
.CONST  007360           1218*   1229*   1960*
.MAMF   005332            555     924    1112    1361    1483*
.PARSR  004472           1331*   1487
.ROTO   007112           1225    1879*
.ROT1   007206           1236    1907*
.TYPE   001030            595     612*
.XOR39  006214           1644    1653*   1693    1774
..3X9   006302           1143    1677*
..8X13  007104           1172    1869*
.22PAR  004750           1398*   1505
.3ISO   006232           1654    1658*
.3IS1   006222           1655*
.3IS9   006250           1657    1659    1663*
.3NOT9  006240           1656    1660*
.3X9    006062           1131    1606*
.8X13   007076           1161    1859*
```

$TYPE   311#

```
ADC      753
ADD      593  614  664  710  728  729  752  797  936  952  983  995 1012 1212 1270
        1295 1305 1318 1321 1324 1419 1446 1548 1581 1594 1595
ASL      748  763  923  981  982  983  984  985  986 1428 1499 1803
BCC     1308 1500 1917
BCS     1889
BEQ      520  541  554  663  666  703  737  739  818  826  955  962  979 1005 1016
        1047 1065 1077 1080 1087 1095 1103 1147 1149 1170 1183 1200 1208 1246 1256
        1279 1281 1283 1347 1399 1532 1557 1568 1612 1642 1654 1656 1659 1691 1697
        1704 1711 1718 1725 1732 1739 1746 1773 1777 1784 1847 1891 1919 1980 1983
        1990
BIC      667  740  751  774  775  988 1169 1978 2047 2053
BIS      669 1498 1563 1984 1986
BIT      665  674  702  738  915 1410 1483 1653 1655 1658 1759 1801 1979 1982 1989
BLT      629
BMI     1662
BNE      617  624  645  675  688  690  760  767  772  809  833  840  848  916  922
         999 1032 1050 1067 1090 1098 1117 1151 1194 1205 1276 1363 1411 1484 1503
        1570 1638 1646 1754 1757 1760 1790 1793 1796 1802 1842 1844 1895 1897 1923
        1925 1967 1969
BPL      563  634  641  671  707 1105 1254 1404 1661 1664 1686 1768
BR       478  631  673  762  823  937  957  971  991 1018 1033 1051 1068 1082 1099
        1153 1211 1213 1287 1290 1352 1373 1657 1985
BVS      997 1063 1497 1804
CLC     1986
CLR      560  655  732  754  770  794  801  853  901  902  905  908  909  911  914
         918 1027 1042 1059 1114 1130 1160 1189 1229 1272 1340 1345 1358 1427 1491
        1515 1536 1537 1539 1550 1562 1585 1608 1684 1698 1705 1712 1719 1726 1733
        1740 1747 1766 1778 1785 1892 1911 2055
CLRB    1991
CMP      662  824  825  921  954  961  998 1015 1035 1046 1052 1064 1075 1079 1086
        1094 1101 1116 1146 1199 1243 1275 1278 1280 1291 1502 1696 1703 1710 1717
        1724 1731 1738 1745 1776 1783 1890 1918
CMPB     623
COM      919  958  959  994 1011 1083 1084 1292 1610 1639 1640 1687 1752 1755 1769
        1780 1781 1787 1788 1791 1794
DEC      759  766  771  808  832  835  847 1031 1049 1060 1066 1097 1193 1204 1637
        1645 1753 1756 1789 1792 1795 1841 1843 1894 1896 1922 1924 1966 1968
DECB     628
EMT      433
HALT     439  477  535  642  653  672  708 1286 1288 1400 1405
INC      562 1028 1043 1089 1091 1209 1210 1242
INCB    1185 1349
JMP      481  656 1249 1262 1362 1401 1452 1805 1848 2057
JSR      469  476  518  555  557  564  595  622  630  648  649  651  676  678  679
         681  684  685  691  694  696  699  700  701  704  730  773  791  799  810
         851  898  899  924  931  940  941  943  951 1024 1040 1072 1112 1113 1118
        1131 1143 1161 1172 1184 1221 1225 1232 1236 1247 1251 1257 1258 1312 1315
        1319 1338 1348 1361 1370 1413 1416 1420 1421 1435 1438 1441 1442 1448 1450
        1577 1607 1644 1678 1683 1693 1699 1706 1713 1720 1727 1734 174. 1748 1763
        1774 1779 1786 1799 1879 1882 1887 1893 1907 1910 1915 1958 1981 2049
MOV      475  480  489  490  491  492  493  494  495  496  501  502  503  504  505
         506  507  508  517  521  522  523  524  525  526  527  529  530  532  534
         539  542  543  544  545  547  548  550  551  552  558  559  592  594  612
         613  619  625  646  647  669  677  683  693  698  709  731  734  735  746
         747  755  761  768  792  793  798  811  820  821  928  829  830  931  934
```

DEMJAB.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

```
                842   844   845   846   849   896   897   903   904   906   910   917   920   929   930
                932   933   934   935   939   946   947   948   949   953   960   972   973   980   987
                990   993   996  1003  1006  1007  1008  1009  1010  1014  1022  1023  1025  1026  1029
               1030  1038  1039  1041  1044  1045  1056  1057  1061  1062  1070  1071  1073  1074  1078
               1085  1092  1093  1106  1111  1126  1127  1128  1129  1134  1135  1137  1139  1140  1141
               1142  1152  1156  1157  1158  1159  1164  1165  1166  1167  1171  1180  1181  1186  1187
               1188  1190  1191  1192  1196  1197  1198  1214  1215  1218  1219  1220  1222  1223  1224
               1230  1231  1233  1234  1235  1240  1241  1255  1267  1268  1269  1271  1274  1277  1294
               1296  1303  1304  1309  1311  1317  1320  1322  1331  1332  1333  1334  1335  1336  1337
               1343  1344  1350  1351  1354  1359  1367  1402  1406  1407  1408  1409  1418  1424  1425
               1426  1440  1444  1447  1485  1486  1487  1488  1489  1490  1496  1501  1504  1505  1506
               1514  1533  1534  1535  1538  1540  1541  1542  1547  1549  1551  1555  1558  1559  1580
               1582  1583  1584  1586  1587  1588  1593  1596  1597  1606  1609  1614  1615  1617  1618
               1619  1620  1622  1623  1624  1625  1627  1628  1629  1630  1632  1633  1634  1635  1647
               1648  1665  1666  1668  1669  1681  1682  1688  1695  1702  1709  1716  1723  1730  1737
               1744  1761  1762  1770  1771  1775  1782  1798  1800  1806  1807  1808  1849  1850  1859
               1860  1869  1870  1880  1881  1883  1885  1888  1898  1899  1908  1909  1913  1916  1926
               1927  1956  1957  1959  1961  1962  1963  1964  1965  1970  1971  1987  1988  2046  2048
               2052  2054
MOVB            616   635   769  1353  1369  1432  1445  1516  1809  1810  1811  1812  1813  1814  1815
               1816  1817  1818  1819  1820  1821  1822  1823  1824  1825  1826  1827  1828  1829  1830
               1831  1832  1833  1834  1835  1836  1837  1838  1839  1840
NEG            1136  1168
NOP             538   566   591   661  1002  1019  1259  1260  1261  1293  1357  1530  1543  1592  1677
RESET          1175  1239  1360  2056
ROL             749   764   765   795   796   802   803   804   805   806   807  1430
ROLB           1429  1931  1932  1933  1934  1935  1936  1937  1938  1939  1940  1941  1942  1943  1944
               1945  1946  1947  1948
ROR             742   743   744   745   756   757   758  1571  1572  1573  1574  1579
RTI             459   567   620   711  1356  1552  1560  1564  1598
RTS             471   597   636   776   812   819   836   841   854  1298  1325  1507  1518  1544  1575
               1589  1649  1667  1670  1951  1861  1871  1900  1928  1949  1972  1992
SEC            1306  1914
SEV            1495
SOB             528   531   546   549   750  1431  1433  1517
SUB             733   827   950   992  1013  1058  1310  1368  1556
SWAB            741  1578
TRAP            432
TST             519   540   553   618   640   643   670   706   736   817   822   839   978  1004  1102
               1104  1148  1150  1182  1201  1203  1207  1273  1282  1307  1346  1398  1403  1531  1567
               1569  1611  1641  1660  1663  1685  1690  1767  1772  1846
TSTB            633   687   689  1253  1365
.ABS            310
.ASCII          569   713   870   871   875   878
.ASCIZ          572   716   717   719   721   722   858   877   879  1383  1386  1463  1465  1468  1470
               1472  1475  1995  2001  2006  2010  2015  2018  2019  2022  2031  2034  2040  2041
.BYTE           605   606   607   609  1341  1342  1520  1521  1522  1523  1524  1525
.END           2060
.ENDC          2060
.EVEN           576   726   862   880  1390  1477  2043
.IF            2060
.LIST           309   439  2060
.MCALL          311
.NLIST          308   439  2060
.REM              1
.REPT           439
```

DEMJAB.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

```
.SSTTL    313     511     575     658     882    1107    1178    1264    1265    1327    1509    1528    1600    1854    1873
.TITLE    312
.WORD     455     456     458     459     461     462     472     473     483     484     533     561     577     578     585
          586     596     610     611     644     813     815     850     856     881     907    1115    1138    1244    1245
         1314    1323    1376    1377    1378    1379    1380    1381    1492
```

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

```
*DEMJAB,DEMJAB.SEQ/SOL.CRF/DS:ERF2/EN:ABS=DSKM:DEMJAB.P11
RUN-TIME:  14 12 3 SECONDS
RUN-TIME RATIO:  53/31=1.7
CORE USED:  8K  (15 PAGES)
```

Spooler runtime 0 Seconds, 37 KCS, 207 disk reads, 5 disk writes, 50 pages